Universitat de Lleida

TREBALL FINAL DE GRAU

ESCOLA POLITÈCNICA SUPERIOR
UNIVERSITAT DE LLEIDA
INSPIRING THE FUTURE

Estudiant: ÒSCAR LLADÓS COS

Titulació: Grau en Enginyeria Electrònica Industrial i Automàtica

Títol de Treball Final de Grau: **DISSENY D'UN PROTOTIP DE CONTROL DE NIVELL DE PINSO I ENVIAMENT DE LA MESURA A UNA BASE DE DADES A TRAVÈS D'UNA TARGETA SIM**

Director/a: **CONCEPCIÓ ROIG MATEU i RAMON VENDRELL RIVERA**

Presentació

Mes: Juliol

Any: 2018

"La información es la gasolina del siglo XXI, y la analítica de datos el motor de combustión"- Peter Sondergaard

Agraïments

Primerament, agrair a l'empresa Corporació Alimentària de Guissona, que m'han donat la oportunitat de realitzar el projecte en aquesta. Especialment al meu tutor, Ramon Vendrell, a l'Albert Vendrell i Josep Sala que en tot moment m'han ajudat i orientat en els moments d'incertesa i problemes.

Paral·lelament també agrair a la meva tutora acadèmica, la Concepció Roig, que des del primer moment m'ha orientat i guiat en el projecte, tot mostrant molt interès per aquest.

Finalment, la família i amics que m'han donat tot el suport necessari per a tirar endavant a pesar de les dificultats.

Sense totes aquestes persones no hauria arribat fins aquí.

Resum del treball

L’empresa Corporació Alimentària de Guissona disposa de gran quantitat de granges per la seva producció carnívora. Aquests animals són alimentats amb els pinsos que la mateixa empresa produeix. Actualment, l’empresa no té control predictiu del nivell a produir de pinso.

Doncs bé, aquest treball està focalitzat en investigar la forma més òptima, eficient i barata de proporcionar les dades necessàries a l’empresa per tal de poder preveure la producció de pinsos, així maximitzant el benefici i minimitzant les pèrdues. Paral·lelament, el granger es podrà beneficiar del sistema també, podrà tenir un control actualitzat del pinso restant en tot moment.

El procès a seguir constarà d’una investigació teòrica tot contrastant opcions, sempre basades en la funcionalitat i pressupostos. Un cop el fonament teòric sigui suficient, serà posat en pràctica, tot realitzant els passos pertinents i necessaris per a cada opció.

Actualment, ja hi ha un prototip inicial, implementat amb Sigfox, i solament està pensat per a ser aplicats a sitjes ja existents. Doncs aquest es considera que es pot optimitzar a nivell de pressupostos i fins i tot buscar alternatives, perquè la situació geogràfica de les granges acostuma a ser remota i l’enviament de dades de forma telemàtica pot causar problemes.

El disseny final hauria de ser compatible per les diverses plataformes d’enviament de dades.

Abstract

“Corporació Alimentària de Guissona” is a big company, which is able to sell meat from their own animals due to the huge range of fields they are competitive at.

This fact could be seen as another way of business, the production of feeds. Currently, they are already producing fodder, even thought they do not have a predictive control about the needs, yet.

This thesis is focused on a main goal, let the company being aware, in real time, of the amount of the fooder left in each farm. If that information could be given, would be possible to keep a control of the production. That is directly related with an increasing of the benefits and a minimization of the loses.

The process to follow will be based on a theoretical investigation, always taking into account the price and the benefits of every single device. When the theoretical knowledge is consistent enough, it will be applied in real life.

Currently, the company has already developed an initial prototype which is only thought to be applied on working silos. In fact, this design can still be improved because the farms are most of the time, located in remote places, with a bad wireless signal. So the purpose of the final design is that it should be able to be switched easily from one transmission method to another.

ÍNDEX

Taula d'il·lustracions

Índex de taules

1. Introducció

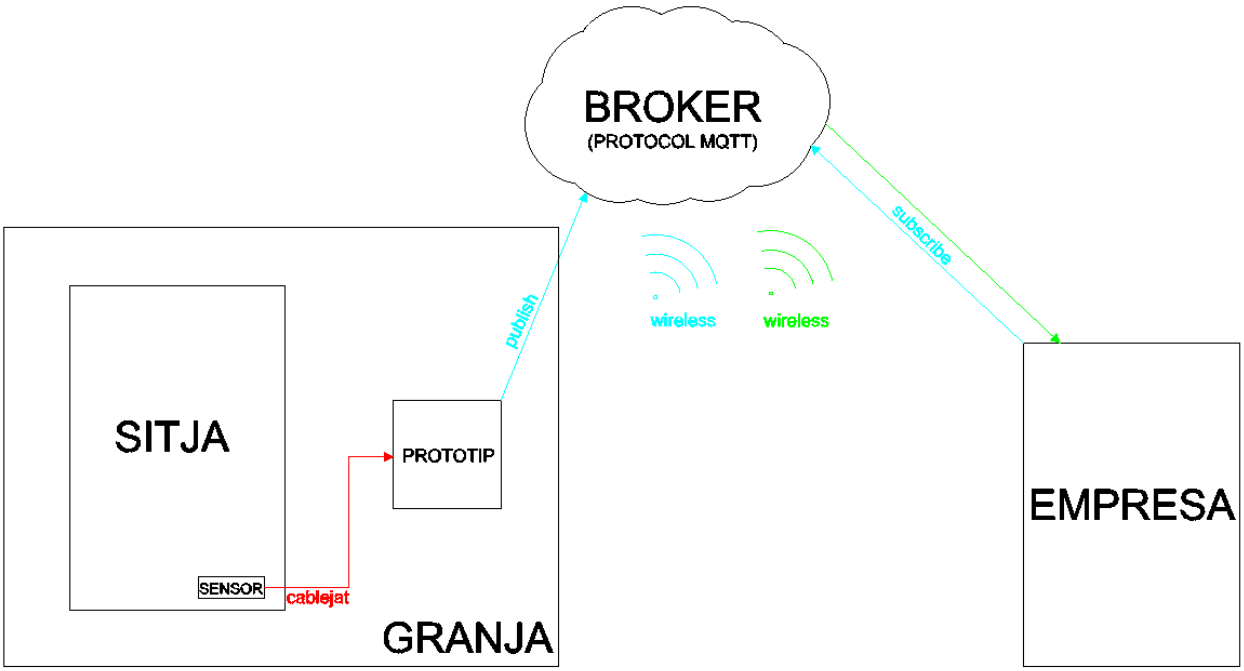
1.1. Descripció del projecte

El present projecte tracta de la construcció d'un sistema intel·ligent de lectura del pinso restant en una sitja, i l'enviament automàtic d'aquesta a l'empresa de forma totalment remota. Així aquestes dades podrien ser utilitzades per controlar la producció del producte. Paral·lelament el granger podria veure la quantitat de pinso en estoc en la sitja depenent del tipus de sistema de mesura empleat.

A priori, el projecte contempla diverses opcions de sistema de mesura i enviament de dades. El problema és que les granges acostumen ha estar ubicades en indrets relativament remots, i amb freqüència hi arriba poca cobertura i senyals dèbils. Així, doncs, es reduiria la probabilitat de no poder obtenir les dades a causa de la geolocalització.

Doncs en el llarg del treball, és valoraran les diverses opcions, i s'arribarà a un prototip final, capaç d'adaptar-se en cada cas concret, al sistema més favorable, depenent de la voluntat del client i les condicions geogràfiques.

En la següent imatge és pot observar un esquema simplificat del projecte en qüestió.



Il·lustració 1: Esquema del projecte amb totes les localitzacions/transmissions.

Es pot observar que hi ha la presència d'una sitja, la qual ha de ser mesurada la seva massa de pinso restant. Aquesta mesura es fa a través d'un sensor que va cablejat directament al prototip a crear en aquest treball. Tant el sensor com el prototip estan pensants per formar part de la granja.

Des d'aquest prototip s'envia les dades a un broker, equivalent amb un espai virtual. I des de l'empresa es pot llegir el que hi ha en aquest espai. Aquesta última comunicació de publicar/llegir el del broker és totalment una connexió sense cable.

Aquest prototip presentarà una placa PCB comuna que en funció de la localització i dels serveis de comunicació disponibles s'utilitzarà un xip o bé un altre. Però sempre mantindrà la seva funcionalitat independentment d'aquestes petites modificacions.

1.2. Antecedents

L'empresa en qüestió s'interessa cada dia més en l'anàlisi i el tractament de les seves dades, altrament dit, el Big Data. Per aconseguir un bon rendiment cal disposar primerament aquestes dades.

L'empresa avui ja té desenvolupades unes certes idees del prototip de lectura de dades en les sitges amb una plataforma anomenada Sigfox. Exactament ha realitzat el codi d'enviament de dades a través del dispositiu característic de la plataforma i també disposa d'algun que altre component que pretén utilitzar en aquest disseny, tot basant-se amb les característiques tècniques que el component utilitzat necessita. La meva finalitat per superar la tasca és aconseguir un nou model compatible i millorat a partir del seu prototip inicial.

1.3. Motivació

La motivació del projecte apareix just quan estava realitzant les meves pràctiques curriculars a l' empresa: "Corporació Alimentària de Guissona".

La pròpia entitat em va proposar participar en un projecte real i actual, que acabaven de començar, i partien d'unes idees molt generals i pràcticament sense desenvolupar. Donant-me així una certa llibertat per tirar endavant el projecte.

Paral·lelament, m'ha agradat molt el tema perquè està estretament relacionat amb l'enginyeria electrònica que he estudiat. Formar part d'un equip d'enginyers electrònics hem permet aprendre moltíssim, tot tirant endavant un projecte real d'una certa magnitud.

1.4. Objectius

Les expectatives a complir d'aquest projecte són:

- Informar-me sobre les diverses opcions de transmissió de dades de forma telemàtica i ser capaç de decidir la millor opció en cada cas concret.
- Explicació acurada del prototip inicial ja elaborat per l'empresa, i l'aprofitament del màxim nombre de components possibles per aconseguir un nou disseny compatible amb aquest.
- Desenvolupar en tots els aspectes necessaris el nou prototip.
- Aconseguir un disseny final amb un cost inferior al comercial, únicament de mesura sense enviament de dades, uns 600 €.
- Estudi de l'impacte que ha tingut el projecte a nivell de producció.

2. Abreviatures i acrònims

M2M: Machine to Machine.

WLAN: Wireless Local Area Network.

LPWAN: Low-Power Wide-Area Network.

WAN: Wide Area Network.

ISM: Industrial, Scientific and Medical.

UNB: Ultra Narrow Band.

SIM: Subscriber Identity Module.

IoT: Internet of Things.

GPRS: General Packet Radio Service.

2G: Second Generation.

3G: Third Generation.

4G: Fourth Generation.

PWM: Pulse-Width Modulation.

PLC: Programable Logic Converter.

LED: Light-Emitting Diode.

RS-232: Recommended Standard 232.

RS-485: Recommended Standard 485.

DC: Direct Current.

AC: Altern Current.

GSM: Global System Mobile.

AT: Attention Command Set.

TCP: Transmission Control Protocol.

IP: Internet Protocol.

Rx: Reception mode.

Tx: Transmission mode.

Bps: Bytes per segon.

F.E.:: Fons d'escala.

HART: Highway Addressable Remote Transducer .

LED: Light-Emitting Diode .

PCB: Printed Circuit Board.

MQTT: Message Queue Telemetry Transport.

RS-232: Recommended Standard 232.

RS-485: Recommended Standard 485.

BLOC I - MARC TEÒRIC

3. Sistemes de comunicació

[0] En aquest apartat es presenten els diferents sistemes de comunicació existents en l'actualitat. En tots ells es fa una breu explicació de les principals característiques, les quals són contrastades amb les necessitats del projecte i es decideix la seva utilització o no.

En l'esquema de la figura 1, equival a la comunicació wireless.

3.1. m2m

Machine-to-Machine (m2m) fa referència a les tecnologies que permeten la comunicació entre màquines. Consisteix en un servei de comunicació, mitjançant xarxes fixes o mòbils, entre una màquina o dispositiu que compleix una determinada funció i una altra màquina o servidor central que controla i monitoritza la primera.

La tecnologia de m2m combina electrònica, telecomunicacions i tecnologies de la informació per a connectar dispositius i sistemes remots que permeten:

- Generar alarmes per fallades o necessitats de manteniment preventiu del mateix dispositiu.
- Generar alarmes sobre comportaments indesitjats en llocs o sobre dispositius.
- Generar informació sobre la situació d'estoc.

Hi ha diferents tipus de tecnologies que ens permeten la connexió m2m, les més comunes són:

- Wireless Local Area Networks (WLAN).
- Bluetooth.
- Lower – Power Wide Area Networks (LPWAN).
- Wide Area Network (WAN).

3.1.1. WLAN

Té un ample de banda gran, de 2,4 GHz a nivell global, i de 5 GHz industrial. Permetent a l'usuari enviar gran quantitat d'informació, però disposa d'un rang limitat, normalment, no pot ser utilitzat a més de 32 metres des del punt d'accés. Per tant, no és l'adequat pel projecte en qüestió ja que s'ha de cobrir una distància de quilòmetres en la majoria de casos per la transmissió de les dades.

3.1.2. BLUETOOTH

Opera amb un reduït ample de banda, de 2,4 a 2,485 GHz i té un curt rang de connectivitat.

No és factible la seva utilització en el projecte.

3.1.3. LPWAN

Típic en àrees de comunicació i sistemes que han d'enviar poca informació però a llargues distàncies.

Cal destacar dues opcions pertanyents amb aquest tipus: - Sigfox i LoRa.

Per tant, ambdues poden ser una opció vàlida pel treball. Posteriorment, seran explicades i valorades.

3.1.4. WAN o CELLULAR

Permet una connexió pràcticament a nivell global i la transmissió d'un nombre elevat d'informació i de gran pes.

Els mòbils en són un clar exemple.

Així doncs, aquestes propietats el fan un sistema molt utilitzat en la comunicació m2m.

També compleix característiques òptimes pel projecte. Serà valorada la seva utilització.

A la il·lustració 2 s'hi poden veure una taula amb les diverses opcions de comunicació. I de cadascuna d'elles hi ha especificades certes característiques bàsiques, juntament amb exemples de cadascuna

CONNECTIVITY OPTIONS

Characteristics of Options

	WLAN	Bluetooth	LPWAN		Cellular
			LoRa	SigFox	
Frequency Band	Unlicensed, global 2.4 GHz ISM	Unlicensed, global 2.4GHz-2.48GHz ISM	WLAN or Cellular	LPWAN or ISM	Radio frequencies
Range	Very limited: Maximum 32 meters	Limited: Maximum 100 meters	Good: Max. 16 kilometers	Good: Max 40 kilometers	Very good: WAN (Wide area network) – tower dependent; reliably strong signal up to 16 kilometers from tower
Unique feature	Ability to transfer large amounts of data quickly	Ability to 'hop' between frequencies	Proven in harsh environments & underground	Very reduced power usage	Long range potential, global connectivity through cell-towers
Drawbacks	Extremely limited range	Limited range Not suitable to transfer large amounts of data	Can't connect to devices as standalone service	Not suitable to transfer large amounts of data	High power consumption
Examples	Smart streetlights Parking meters Smart home power meters	Audio & mobile apps Wearables Smart home security sensors	Smart agriculture Sensor networks	Remote monitoring systems Alarms	Logistics – asset tracking Transport – keyless locking Smart metering applications

Il·lustració 2: Quadre resum de les opcions de connectivitat amb propietats a destacar. (Font: Emnify-Whitepaper.pdf)

3.2. Plataformes de transmissió de dades

En aquest punt s'explica de forma detallada totes i cadascuna de les opcions de transmissió de dades possibles d'acord amb el sistema de comunicació explicat prèviament. També és valora la seva possibilitat d'aplicació en el projecta i s'argumenta la decisió.

L'equivalent a l'esquema de la figura 1 és el broker, que és el lloc d'emmagatzematge provisional de les dades.

3.2.1. Sigfox

[1] Sigfox és una empresa francesa de xarxa sense fil que permet connectar objectes i dispositius que tenen la necessitat d'estar connectats de forma permanent i enviar un nombre reduït de dades a una base de dades de la mateixa empresa.

Il·lustració 3: Logotip companyia Sigfox.

Per realitzar aquesta connexió en xarxa és necessari incorporar un xip que sigui compatible amb la mateixa i la seva pertinent programació.

Sigfox ha apostat per una xarxa petita, amb un ample de banda reduit, però suficient per permetre enviar missatges de 12 bytes, durant 140 vegades al dia i 6 vegades per hora. Permet descarregar 4 a l'usuari enregistrat missatges de 12 bytes al dia. També disposa d'interessants avantatges per la comunicació dels objectes connectats:

- Eficiència energètica, amb la capacitat d'allargar la bateria d'alguns dispositius fins a 15 anys.
- Utilització de freqüències lliures (ISM) resistents a interferències.
- Connectivitat UNB bidireccional.
- Gestió senzilla i basada en el núvol. Es poden controlar tots els dispositius connectats a la xarxa des d' un únic lloc.
- Cobertura en 14 països, entre ells gran part d'Espanya.

La densitat de les cèl·lules en la xarxa Sigfox es basa en un rang intermedi de uns 30-50 Km en zones rurals. En zones urbanes la densitat pot reduir-se a 3-10 Km a causa dels obstacles i major soroll de distorsió. Amb els nodes a l'aire lliure s'ha aconseguit distàncies de més de 1000 quilòmetres, el que el fa ser adequat per a zones remotes i despoblades.

Sigfox pot actuar com a complement a xarxes ja existents o actuar de forma independent. Els detectors i xips que poden incloure són de mida reduïda.

També s'ha de tenir en compte, que Sigfox emmagatzema les seves dades a una base de dades de la companyia, per tant, no és necessari la creació d'una base pròpia.

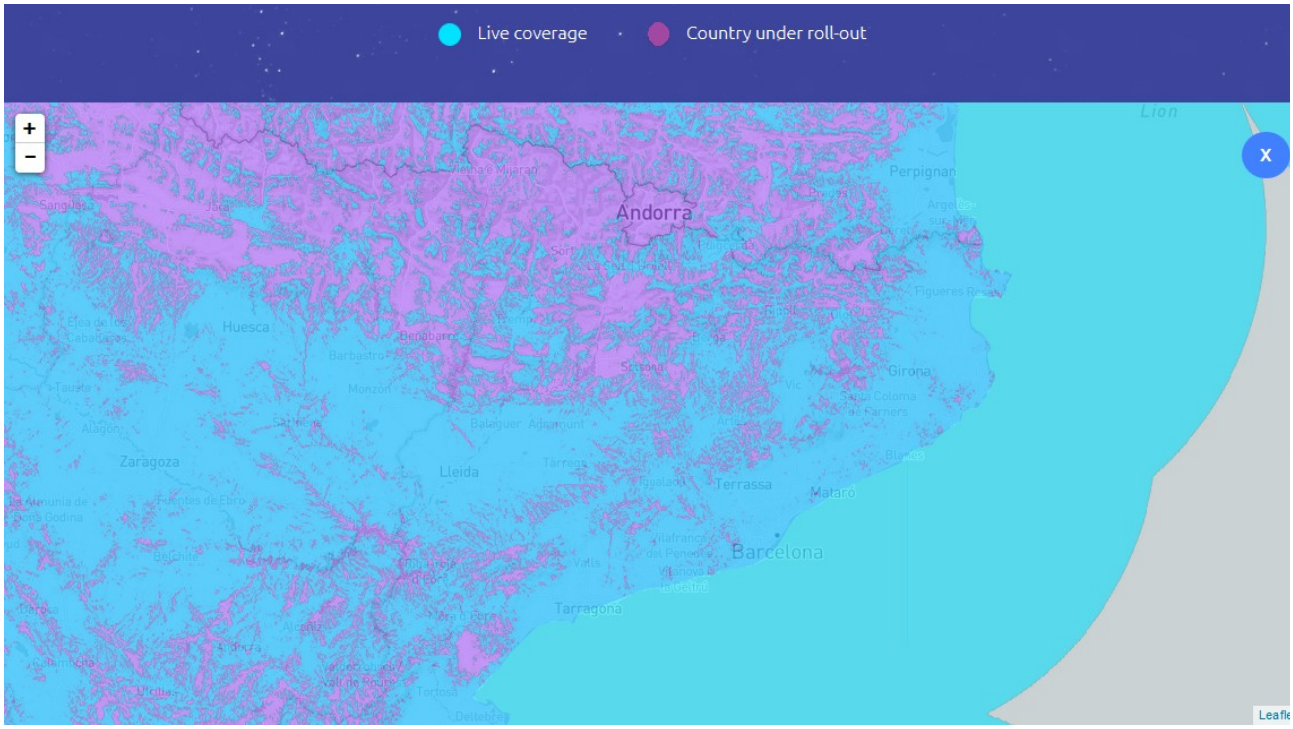
Per tant, la plataforma Sigfox proporciona totes les característiques idònies per a establir una comunicació entre una zona rural i l'empresa, juntament amb una gran autonomia i un preu molt assequible d' $\frac{1\,€}{any{\cdot}usuari}$

La principal preocupació del sistema és que depenent de la localització de la granja en qüestió, podria no rebre senyal de la companyia. Per poder tenir una idea inicial de la possibilitat d'aplicar aquesta transmissió de dades, la mateixa companyia ens proporciona un servei en línia que ens permet saber la possibilitat d'utilització de la plataforma depenent de ubicació a nivell mundial.

Aquest servei es troba en el següent link:

https://www.sigfox.com/en/coverage

A la il·lustració 4 es pot observar una captura de pantalla del mapa de cobertura a la regio de Catalunya i Andorra. Amb color blau les zones totalment cobertes i amb lila les zones en desenvolupament que pròximament tindran cobertura.



Il·lustració 4: Mapa cobertura Sigfox a Catalunya. Lila: les zones amb desenvolupament. Blau: les zones amb cobertura.

3.2.2. LoRa

[2][3] LoRaWAN (acrònim de Long Range Wide-Area network) és una xarxa de gran abast i de baix consum.

És interessant perquè:

- Consumeix molt poc, la seva bateria té una vida entre 5 i 10 anys. Per aconseguir això, s'activa de forma periòdica, solament quan és necessari.
- Té un gran rang d'arribada que permet accedir en els indrets més remots i complicats.
- És una xarxa bidireccional.
- La velocitat d'enviament de dades va des de 0,3 kbps a 50 kbps.
- Pot ser utilitzada en llocs interiors, perquè pot penetrar parets. A menor freqüència, major poder de traspassar material.
- Treballa a 868 MHz, que és una banda de freqüència lliure i gratuïta.
- També ens permet realitzar una geolocalització externa sense GPS.

A diferència de amb Sigfox, que es depèn de la cobertura que el servei proporciona, amb LoRa, si aquesta cobertura no és existent, el mateix usuari pot crear un punts d'accés en forma de Gateways, mitjançant elements electrònics com són exemples l'Arduino o Raspberry.Pi, creant així una xarxa il·limitada de connexions.

Aquesta necessita una base de dades pròpia, ja que l'empresa en si, no la proporciona.

Per tant, presenta també unes característiques excel·lents per l'execució del projecte i l'obtenció d'unes visions de futur molt elevades en cas que l'empresa es planteges una instal·lació de LoRa a gran escala cobrint certs territoris. Desafortunadament, avui en dia encara no es factible aquesta opció.

Il·lustració 5: Logotip companyia LoRa.

3.2.3. m2m SIM Card

[4] Machine-to-Machine SIM (m2m SIM) fa referència a la tecnologia que permet la comunicació entre sensors i diversos aparells, tot comunicant-se entre ells, formant part del conegut "Internet de les coses" (IoT).

Les m2m SIM són utilitzades per una comunicació - entre dispositiu i sistema de control, base de dades o aparell - de dades. Aquesta targeta SIM s'ha d'implementar en l'aparell emissor de dades, el qual, enviarà aquesta informació al servidor principal, permetent així a la companyia, tenir la informació desitjada. Perquè aquesta pot ser extreta de llocs remots, únicament és necessàri que hi hagi una cobertura suficient, de la companyia contractada.

Per part de l'empresa, hauria de crear o subcontractar una base de dades per emmagatzemar tota la informació enviada des de les diverses granges, ja que el servei de connexió SIM card, únicament facilita la transmissió de dades i no l'emmagatzematge d'aquestes.

Aquestes targetes SIM tenen gran quantitat d'utilitzacions industrials. Tenen una capacitat de transferir informació i poden realitzar molts cicles (més de 8 milions), per tant, el cicle de vida és molt superior que el de les SIM utilitzades en els mòbils pels usuaris.

Pel que fa al material de fabricació d'aquestes targes, és molt resistent als agents de temperatura, estan preparades per tenir un excel·lent funcionament entre -40 ºC i 105 ºC. Totes les característiques en conjunt els hi dona una vida útil aproximada d' uns 15 anys.

Pel que fa a l'enviament de dades i la seva velocitat, disponibilitat de cobertura...dependrà totalment del tipus de quota que es contracti a la pertinent companyia telefònica. A escollir entre: GPRS, 2G, 3G i també 4G. Concretament, a l'empresa, Movistar, el servei es subministraria amb GPRS per uns $0{,}7\ \frac{€}{mes*usuari}$.

Il·lustració 6: Logotip m2m-SIM card companyia Movistar.

3.3. Plataformes adecuada pel projecte.

Després de fer un estudi de cada plataforma i tenint en compte les característiques de cadascuna d'elles, i comparant-les amb les del projecte que són:

- Granges situades en espais remots.
- Llargues distàncies d'enviament de dades.

Clarament, les úniques opcions viables són:

- Sigfox.
- m2m SIM card.

Pel que fa al LoRa ha estat descartat ja que necessita d'un conjunt de dispositius que actuïn com a antenes de transmissió de senyal. Això suposa un gran cost per l'empresa perquè necessitaria una instal·lació massiva d'antenes a gran quantitat de localitzacions per cobrir tot el territori que necessiten controlar.

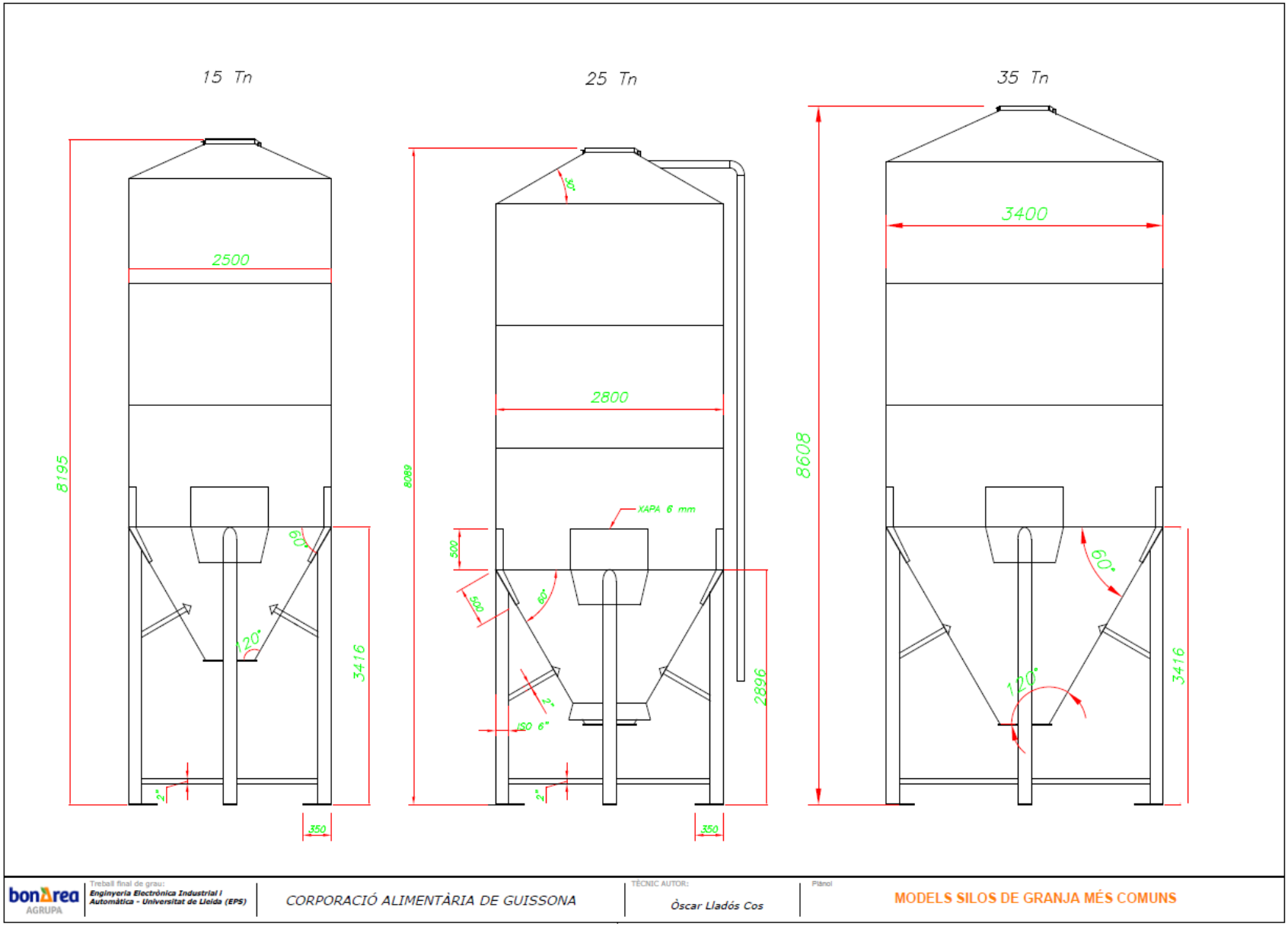
Per tant, el projecte és centrarà únicament amb fer un prototip de dispositiu capaç d'enviar dades tant amb Sigfox com amb m2m SIM card.

BLOC II - ELABORACIÓ DEL PROTOTIP

4. Dissenys més comuns de les sitges fabricades

En aquest apartat hi ha una presentació dels dissenys de sitges més comunes fabricades per l'empresa juntament amb les cotes pertinents i necessàries per la seva elaboració.

En l'esquema general de la figura 1 és la sitja que presenta el sensor.



Il·lustració 7: Models de les sitges de granja més comunes.

5. Sensors a utilitzar

En aquesta secció s'estudia els diferents tipus de sensor que poden ser utilitzats per la mesura del pinso restant en la sitja.

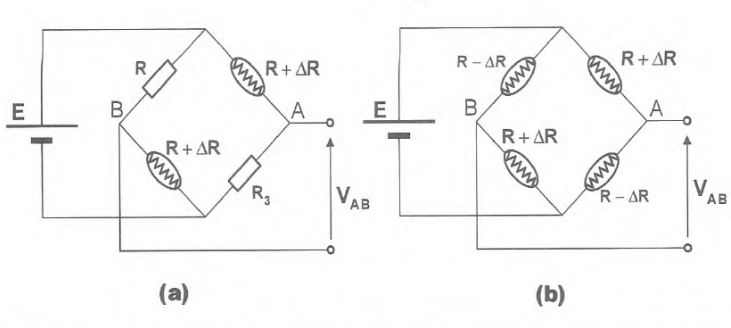
En l'esquema de la figura 1 és el sensor que es troba a la sitja. Aquest anirà cablejat directament cap al prototip que posteriorment, s'explicarà el seu disseny.

5.1. Cèl·lules de Carga

[5]Les cèl·lules de carga són un tipus de sensors resistius que mesuren els esforços mecànics en materials en els que, entre els seus extrems, es modifica la resistència amb l'esforç aplicat sobre aquest. Doncs tècnicament s'anomenen "galgues extensomètriques".

Internament, circuits emprats de manera interna i basats amb el Pont de Wheatstone.

En la següent figura es mostra un pont de Wheatston amb 2 i 4 galgues extensomètriques.



Il·lustració 8: Pont de Wheatstone. a) Amb dues galgues extensomètriques. b) Amb quatre galgues extensomètriques.

Des del punt de vista teòric el pont de Wheatstone amb dues galgues extensomètriques; podem saber el seu valor de voltatge de sortida, a partir del qual, es pot quantificar el pes a partir d'un calibratge posterior.

$$Vab = \frac{-k \cdot \varepsilon}{2} \cdot E$$

I si el pont és amb 4 galgues, l'equació de sortida és la següent:

$$Vab = -k \cdot \varepsilon \cdot E$$

on ε és l'elongació per unitat de longitud de la galga extensomètrica.

$$\varepsilon = \frac{\Delta l}{l} = \frac{F}{E \cdot A}$$

F és la força que s'aplica a la galga.

E és el mòdul de Young, propietat de cada material en concret.

A és l'àrea de la secció del cable.

I pel que fa a la k és el factor de galga, que determina la sensibilitat de la resistència als canvis de longitud.

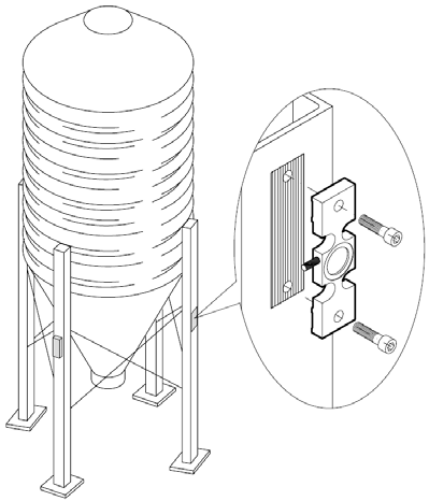
$$k = \frac{\frac{\Delta R}{R}}{\frac{\Delta}{l}}$$

5.1.1. SV-3000

[6]Aquest tipus de sensor està dissenyat per a treballar tant en tracció com en compressió. Especialment està pensat per a mesurar massa en funció de deformacions estructurals de l'element on es troba situat.

Així doncs, té una precisió molt acurada, fet que implica que la seva deformació sigui molt petita, concretament, de $3000\mu \cdot \varepsilon$.

Pel que fa al muntatge d'aquest, és relativament fàcil. Consisteix en incorporar de forma permanent la cèl·lula de carga en la biga de la sitja. Aquesta realitza una deformació diminuta en funció de la carga que presenta. Doncs, aquesta lleu variació pot ser quantificada a través de la cèl·lula de carga en qüestió. A posterior-hi, es pot tractar aquest valor i establir la relació pertinent amb la massa de pinso restant en cada moment.

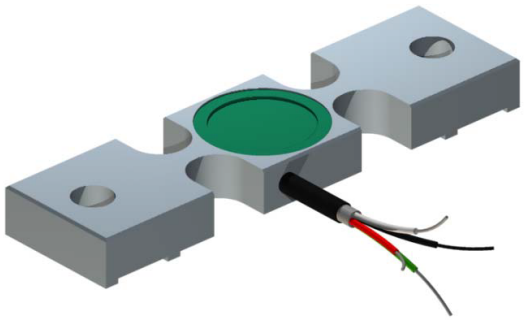


Il·lustració 9: Col·locació de la cèl·lula de carga a la sitja.

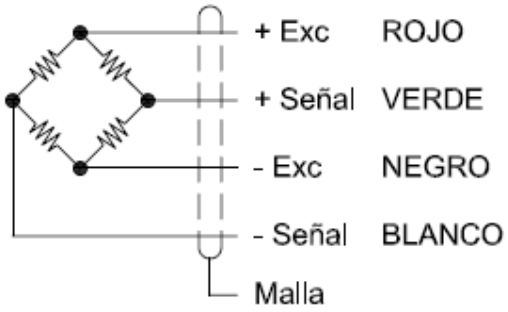
Les característiques tècniques bàsiques d'aquesta cèl·lula de carga són:

- Sensibilitat de 2 mV/V a 3000 με.
- Una tolerància a la sensibilitat de +- 20 % F.E..
- Un màxim voltatge d'excitació de 12 V.
- Una precisió de 0,2%.
- Rang de temperatura de servei entre -20 i 60 ºC,

Les següents figures són un exemple del tipus de cèl·lula de carga proposada, juntament amb el seu pertinent esquema de connexió.



Il·lustració 10: Cèl·lula de carga SV-3000.

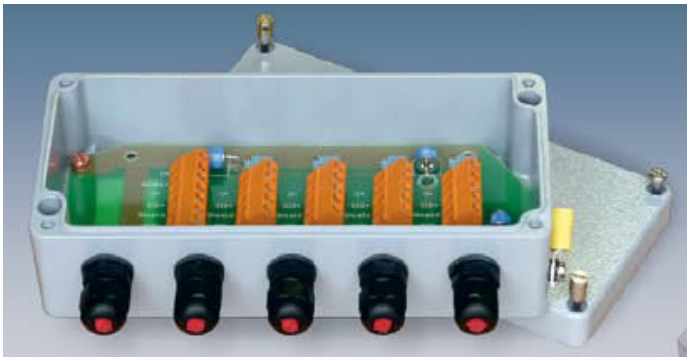


Il·lustració 11: Esquema de connexió de la SV-3000.

5.1.2. Caixes Suma Uticell de 4 cèl·lules de carga

[7]Aquest tipus de caixes permeten la connexió de fins a 4 cèl·lules de carga al indicador de pesatge.

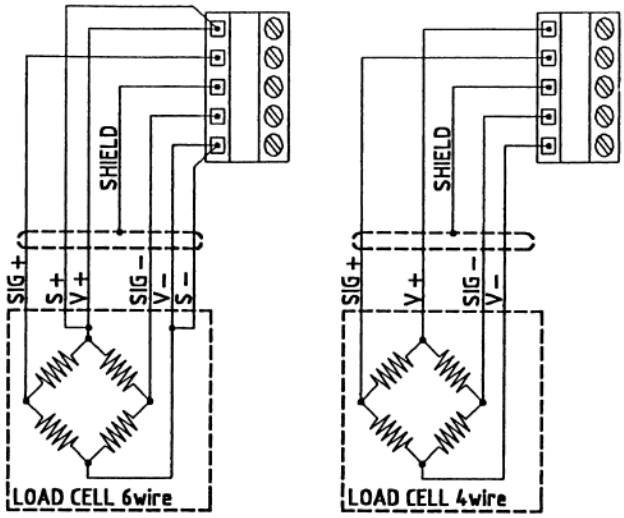
D'aquesta manera es poden connectar totes les cèl·lules de carga necessàries per a cada pota de la sitja i obtenir un únic valor de sortida. A partir d'aquesta ja es pot obtenir una mesura acurada de la massa restant en el recipient mesurat.



Il·lustració 12: Caixa Suma d' Uticell de 4 cèl·lules de carga.

I pel que fa a la seva connectivitat són compatibles amb cèl·lules de carrega tant de 4 com de 6 fils.

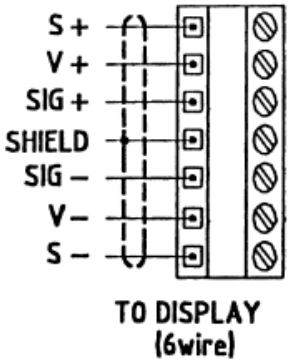
Però cada tipus concret a de seguir el seu esquema de connectivitat que és el següent:



Il·lustració 13: Connectivitat d'una cèl·lula de carga de 6 i 4 cables a la caixa Suma.

I pel que fa a la sortida de la caixa Suma, pot ser directament connectada al lector de pes en qüestió.

El seu esquema de sortida amb la pertinent nomenclatura és:



Il·lustració 14: Esquema de sortida de la caixa Suma.

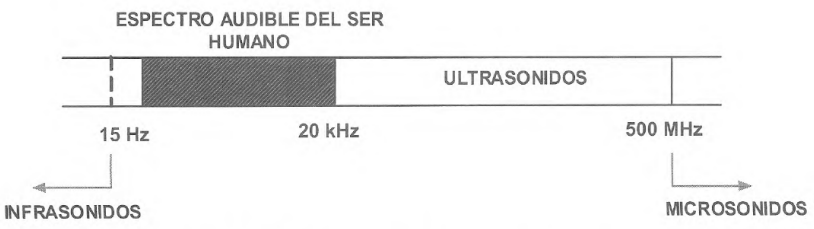
Per tant, independentment del tipus de cèl·lula de carga que es connecti, la sortida de la caixa Suma serà comuna.

5.2. Sensors Ultrasònics i Radars

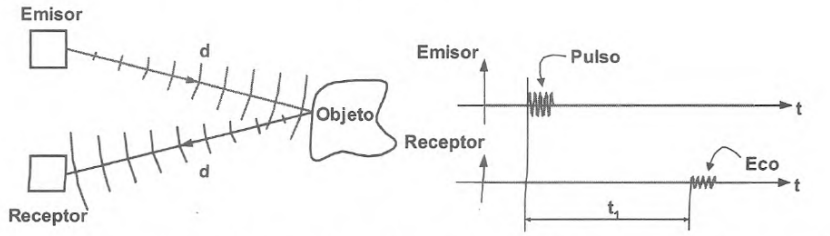
[5] Aquest tipus de sensors permeten captar ones de pressió en un medi elàstic i generar el corresponent senyal elèctric de sortida. Doncs aquest tipus de sensors tenen una reversibilitat ja què ells mateixos són capaços d'enviar el ultrasò o bé una ona electromagnètica i de rebre'l, per tant, són emissors i receptors al mateix temps.

Aquest temps entre enviament i arribada es pot mesurar. Un cop el tenim, a partir d'equacions matemàtiques bàsiques es pot aconseguir una aproximació molt acurada de la distància. Això s'anomena tècnica de mesura de distàncies per impuls-eco.

Els ultrasons és troben entre 20 kHz i 500 MHz.



Il·lustració 15: Espectre de vibracions acústiques.



Il·lustració 16: Funcionament bàsic de la tècnica de mesura de distàncies impuls-eco.

És a dir, si el temps entre el pols i l' eco rebut es t_1 i la velocitat del ultrasò o ona electromagnètica en el medi de propagació és c , la distància del conjunt emissor-receptor i objecte serà:

$$d = \frac{c \cdot t_1}{2}$$

Una vegada tenim la distància fins al producte, si sabem la forma del recipient que el conté, podem saber el volum, per tant també la massa de producte emmagatzemat.

És a dir, es necessita un processat matemàtic extra.

5.2.1. Vega Puls 69

[8]És un sensor per la mesura contínua de nivell de sòlids a granel en les condicions més variades del procès. És ideal per la mesura en sitges molt altes, tolves grans i dipòsits segmentats. Presenta una concentració focalitzada de la senyal, donant una precisió acurada en la mesura, evitant els rebots amb els laterals del recipient de contenció.

La seva col·locació en la sitja és a la part superior d'aquesta. No necessàriament ha d'estar totalment centrat ja que presenta una joc d'angle de col·locació, així aconseguint focalitzar el màxim la senyal i evitar errors.

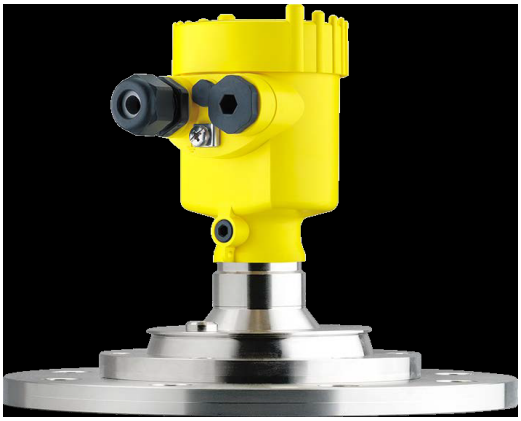
Els seus principals avantatges són:

- No necessita manteniment gràcies al mètode de mesura sense contacte.
- El seu desgast es mínim.
- Presenta un sistema de mesura d'eliminació de partícules flotants, així evitant errors elevats.

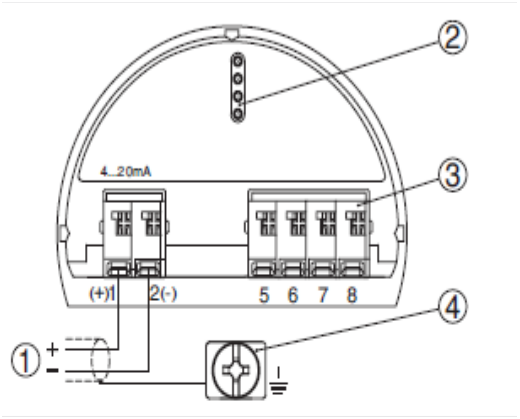
Les seves característiques tècniques són:

- Rang de mesura fins a 120m.
- Error de mesura de $\pm 5\,mm$.
- Capacitat de treball amb intervals de temperatura: entre -40 ºC i 200 ºC.
- Tensió de treball: 12-35 V DC
- Pot ser de 2 o 4 fils, i ambdues tenen senyal de sortida de 4-20 mA / HART

Il·lustració 17: Posicionament Vegapuls 69.



Il·lustració 18: Vega Puls 69



Il·lustració 19: Connexió elèctrica Vega Puls 69. 1) Alimentació de tensió. 2)Adaptador d'interfase. 3)Configuració externa. 4) Posada a terra.

Per tant, si retorna la distància, significa que s'ha d'aconseguir obtenir una relació entre la massa de pinso restant i la distància des de la part superior al material. Doncs aquesta relació s'obté de forma matemàtica en apartats posteriors.

Això és un inconvenient important, del tipus de sistema de mesura a gran escala. No totes les sitges tenen les mateixes dimensions, per tant cada cas concret s'ha de particularitzar.

Aquest sensor pot ser una opció vàlida per a certs casos determinats, però el projecte es focalitzarà en les cèl·lules de cargues que és un sistema de mesura més genèric.

5.3. Valoració dels sensors

Un cop vist els fonaments teòrics i les característiques del model escollit per a cada tipus de sensor, en aquest apartat es fa una valoració genèrica d'ambdós.

Pel prototip final els dos sensors són totalment vàlids i funcionalment correctes. Però el sensor radar/ultrasònic necessita d'un càlcul matemàtic de la massa de pinso en funció de la distància. Per tant, està directament relacionat amb les dimensions de la sitja i aquestes són de diferents models com ja s'ha mostrat anteriorment. Per tant, per a cada sitja el codi del microcontrolador ha de ser modificat.

Aquest problema pot ser totalment evitat amb les cèl·lules de carga, que poden ser calibrades de forma manual sense necessitat de modificar codi. Permeten així tenir un codi genèric per a tots els casos.

6. Components utilitzats

Aquesta secció presenta els diferents components que seran utilitzats pel prototip final. Es presentarà tant els components que són comuns per ambdues plataformes proposades, com els components característics per a cada una.

Aquests components tots ells formaran part del que l'esquema de la figura 1 s'ha anomenat com a prototip.

6.1. Components característics del prototip amb Sigfox

6.1.1. Telit LE51-868 S

Aquest component és un xip que serveix per gestionar la comunicació del microcontrolador cap a la central. En l'esquema de la figura 1 està situat a l'interior del prototip.

[9]És un mòdul tant de curt com llarg rang, dissenyat per treballar entre les freqüències de 863-870 MHz. Treballa seguint els protocols de l'empresa Telit i actuant com a port de comunicació amb la companyia Sigfox.

Aquest tipus de mòdul ens permet un gran rang de connexions sense fil entre mòduls i equips preparats per la comunicació.

Pel que fa a la seva estructura és de dimensions reduïdes, i es totalment adaptable amb altres estructures electròniques. Donant així una infinitat de possibilitats i aplicacions on pot ser incorporat. Aquest permet una comunicació amb Sigfox, sempre que arribi senyal del propi servei.

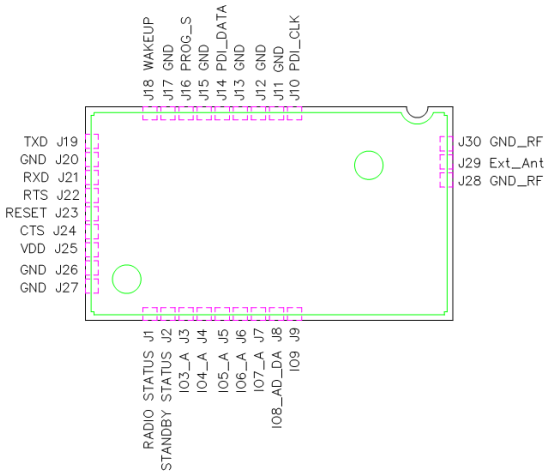
Permet enviar 140 missatges de 12 bytes per dia i ho fa a una velocitat de 100 bps.

Com a característiques elèctriques cal destacar i tenir en compte:

- Font d'alimentació: ha d'estar entre 2,0 i 3,6 V DC
- Els consums del dispositiu a 3V:
  - Amb standby: inferior a 2 μA.
  - Amb wake-up (interrupt): 1 μA

Les dues imatges següents són el mòdul integrat del microxip explicat juntament amb el seu Pin-layout.

Il·lustració 20: Telit LE51-868S.



Il·lustració 21: Telit LE51-868S Pin layout.

6.2. Components característics del prototip m2m SIM Card.

6.2.1. Telit GL865-QUAD

Aquest component és un xip que serveix per gestionar la comunicació del microcontrolador cap a la central. En l'esquema de la figura 1 està situat a l'interior del prototip.

[10] És el nou representant de superfícies integrades de Telit, capaces de transmetre i rebre informació a traves de GSM i GPRS.

El component, presenta un diminut consum, una gran funcionalitat en un elevat rang de temperatures i un disseny compacte. La circuiteria interna està realitzada amb un disseny de 4 capes de PCB.

És un dispositiu adequat per projectes de no molta complexitat, de cost reduït tot i que de gran varietat, perquè ens permet l'adaptació d'aquest en cada cas determinat a partir de les connexions voluntàries dels pins.

Presenta també una interfície d'interpretació del llenguatge Python i C++, permeten una fàcil programació per a solucions m2m. Pel que fa a les comandes d'enviament de dades a traves de la targeta SIM es realitzen a través de comandes AT.

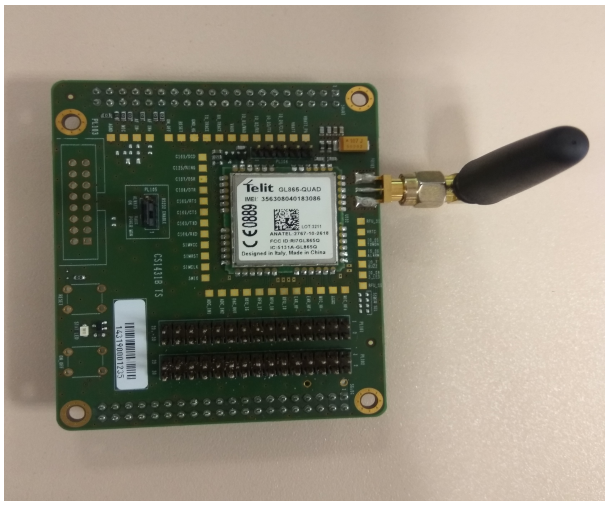
Les principals característiques del component són:

- Quadruble banda GSM/GPRS 850/900/1800/1900 MHz.
- Port sèrie multiplexor 3GPP TS27.020.
- Accés a targeta SIM.
- Accés a TCP/IP a través de comandaments AT.

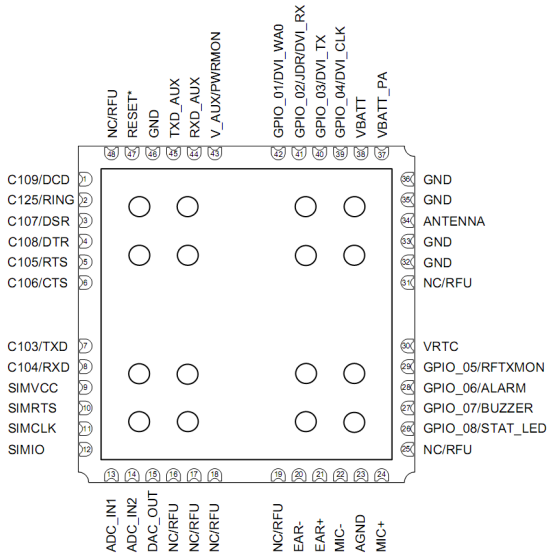
Les característiques elèctriques a destacar i tenir en compte són:

- Voltatge de sortida:
  - Classe 4 (2W) a 850/900 MHz.
  - Classe 1(1W) a 1800/1900 MHz.
- Voltatge d'alimentació: entre un rang de 3,22-4,5 DC però el recomanat és de 3,8 V DC.
- Consum del component:
  - Aparell sense funcionament: <5 μA.
  - En funcionament; 1,5 mA.

Les dues imatges següents són el mòdul integrat del microxip explicat juntament amb el seu Pin-layout.



Il·lustració 22: Telit GL865-QUAD.



Il·lustració 23: Telit GL865-QUAD Pin layout.

6.2.2. SIM900 GSM Shield

Com a característiques principals del component cal destacar:

- Voltatge de sortida:
  - Clase 4 (2 W) 850 / 900 MHz.
  - Clase 1 (1 W) 1800 / 1900MHz.
- Accés a TCP/IP.
- Consum de 1,5 mA
- Voltatge d'entrada: entre 3,4 i 4,5 VDC.
- Controlat amb comandes AT.

Il·lustració 24: SIM900 GSM Shield.

[11]El prototip final està pensat per a ser realitzat amb el component anterior. A pesar d'això pel disseny del prototip en targeta SIM s'utilitzarà aquest nou component. El motiu és la facilitat de comunicació amb l' Arduino pel que fa a la connectivitat de pins i la comoditat en l'hora de fer proves.

Pel que fa a la compatibilitat de codi intern del microxip, és totalment idèntic utilitzant ambdós components. Els pins a connectar amb el microprocessador també són els mateixos. Per tant, aquesta total reversibilitat fa indiferent la seva utilització. Tot i que a nivell industrial és mes fiable el GL865.

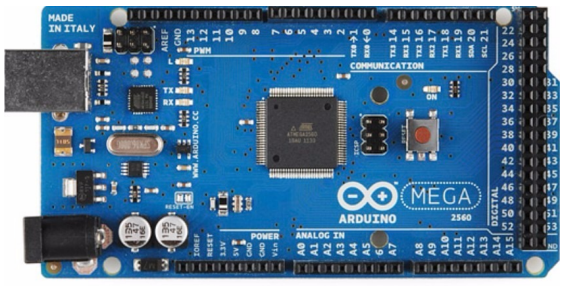
6.3. Components comuns entre prototips

6.3.1. ATMEGA 2560 16U

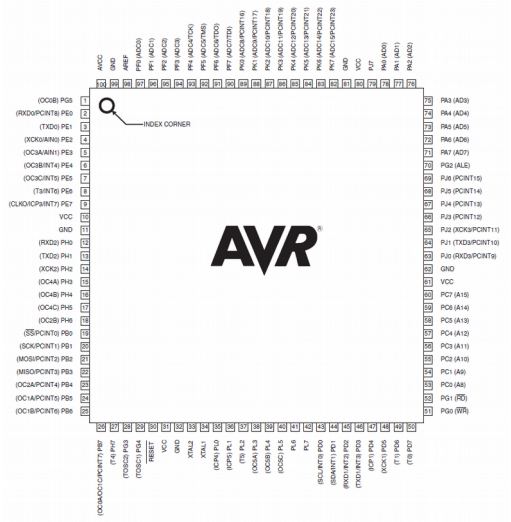
[12]És un microxip que presenta:

- 54 entrades/sortides digitals, de les quals 15 poden ser utilitzades com a sortides PWM.
- També presenta 16 entrades analògiques.
- Té un oscil·lador de cristall de 16 MHz.
- 2 Timmers/Comptadors de 8 bits amb mode de comparació.
- 4 Timmers/comptadors de 16 bits amb mode de comparació i de captura.
- Sortida de comparador modular.
- 4 Serial USART programables.
- Consum molt baix:
  - Mode actiu: 1MHz, 1,8 V: 500 µA.
  - Mode inactiu: 0,1 µA a 1,8 V.
- Grau de velocitat: 0-16 MHz a 4,5 V-5,5 V.

Pot ser programat amb el llenguatge característic de l'Arduino, el c++. De fet aquest és el xip que està integrat en l'Arduino Mega 2560. Durant la realització del prototip , es farà amb una placa Arduino, per la facilitat de connexió que facilita la placa. Un cop el disseny sigui definitiu, s'implementarà amb únicament amb el xip.



Il·lustració 25: Arduino Mega 2560.



Il·lustració 26: Pin layout microxip Mega 2560.

6.3.2. SWIFT RAIL RS

[13] És un indicador de pesatge i transmissor de dades d'alta velocitat, ideal per a aplicacions de pesatge estàtic i dinàmic. Les seves versions per a muntatge en carril DIN o en el Panell el fan especialment indicat per a les aplicacions de pesatge en processos industrials i de maquinària.

Presenta diverses possibilitats de transmissió de dades, les seves entrades i sortides digitals ens permeten la fàcil connexió amb els PLC, ordenadors i els sistemes remots, en els principals estàndards de comunicació industrial.

Com a característiques principals presenta:

- Un display de 6 dígits LED de 10 mm.
- Una alta velocitat. Permet realitzar 2400 lectures/segon.
- Permet un calibratge amb masses o en mV/V.
- Connexió de 8 cèl·lules de carrega de 350 Ω o bé de 16 de 700 Ω.
- Comunicació RS-232 i RS485 amb opció a Profibus, Profinet o Ethernet/IP.
- Sortida analògica configurable 0-5 V, 0-10 V, 0-20 mA i 4-20 mA amb separació galvànica.
- 3 Entrades digitals amb un led d'estat.
- 3 sortides digitals amb un led d'estat.
- Alimentació externa de 10-28 V DC.

Per tant, aquest model es pot utilitzar juntament amb cèl·lules de carga per al pesatge, enviar aquest valor a través dels ports de comunicació que presenta cap al microcontrolador, el qual enviarà les dades al servidor a posterior-hi.

Il·lustració 27: Mòduls del Swift Rail RS.

6.3.3. DR-15-24

[14] El Swift Rail ja explicat, necessita una alimentació externa continua d'entre 10-28 V, per tant, necessitem incorporar un transformador d'alterna a continua i reduir aquest voltatge a un valor comú, com podria ser 24 V.

Per realitzar aquesta tasca, hi han diverses opcions comercials, molt assequibles de preu i totalment recomanables. En aquest cas s'ha utilitzat el DR-15-24 de la marca Mean Well.

Les seves principals característiques tècniques són:

- Tipus d'entrada: 1 fase.
- Voltatge d'entrada: 85 VAC-264 VAC o 120 VDC-370 VDC.
- Voltatge de sortida: 24 VDC
- Corrent de sortida: 0,63 A
- Numero de sortides: 1
- Dimensions: 56x93x25 mm

Il·lustració 28: DR-15-24.

6.3.4. RS-15-3,3

[15] Tal i com en les característiques tècniques del mòdul de comunicació amb Sigfox, Telit, s'ha especificat, el voltatge d'alimentació es de 3,3 V DC. Per tant, hi ha una clara necessitat d'un altre transformador de corrent contínua a alterna per adequar les característiques d'entrada.

També trobem opcions comercials molt vàlides i assequibles de preu.

El convertidor utilitzat és: RS-15-3,3 de Mean Well.

Les característiques tècniques que el fan òptim són:

- Voltatge d'entrada: 85 VAC-264 VAC o 120 V DC-370 V DC.
- Tensió de sortida: 3,3 V DC.
- Corrent de sortida 3A.
- Dimensions: 62,5x28x51 mm.

Com a garanties de seguretat:

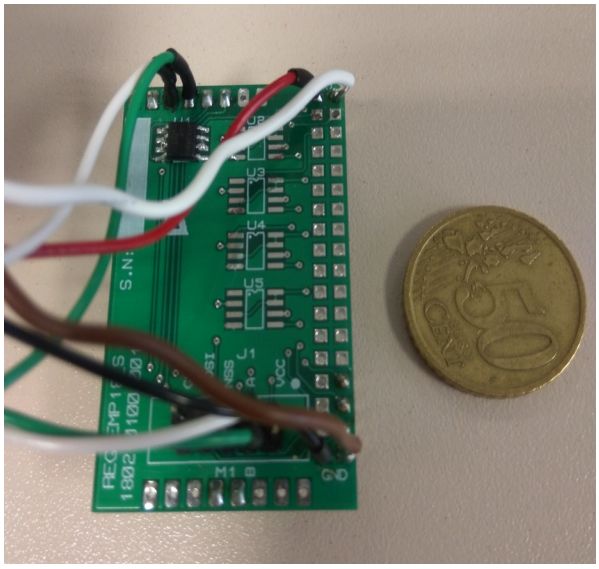
- Proteccions de curtcircuit, sobrecarrega, contra pics de voltatge i de temperatura.
- Sistema de refrigeració amb aire.
- Gran capacitat d'operar a un rang elevat de temperatures fins a uns 70 ºC.

Il·lustració 29: DR-15-3,3.

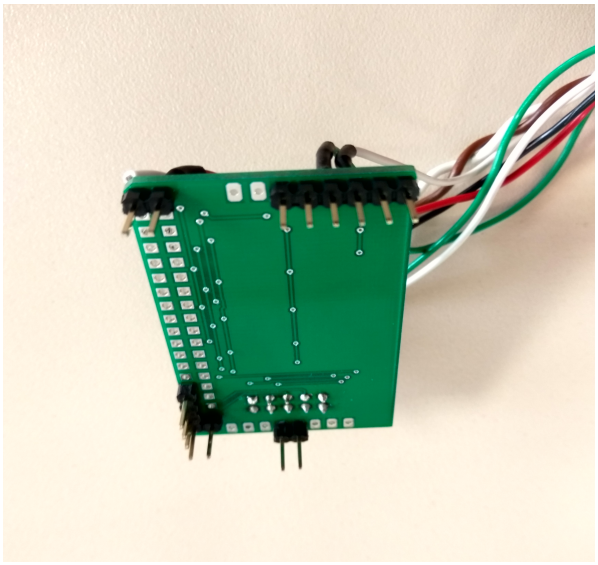
6.3.5. Placa connexió entre components i l' Arduino.

Aquesta placa electrònica, creada durant el disseny del projecte té una finalitat provisional i ens permet establir una fàcil comunicació entre l' Arduino, que és el microcontrolador, i el component emprat per enviar dades.

Per tant, és totalment provisional, emprada per les proves de disseny, ja què un cop aconseguit l'objectiu, tal i com ja ha estat esmentat en altres parts, és realitzarà un únic circuit integrat en PCB que comprendrà entre altres elements, aquesta placa.

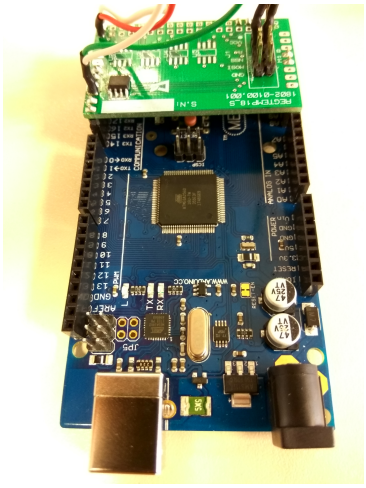


Il·lustració 30: Placa adaptada a Arduino, part superior.



Il·lustració 31: Placa adaptada a Arduino, part inferior.

Doncs la seva forma de connexió amb el microcontrolador Arduino és la següent:



Il·lustració 32: Placa d'adaptació connectada a l' Arduino.

Per tal de poder fer més gràfica l'explicació d'aquesta i la seva funcionalitat, primer cal descriure un dels principals components, el MAX-3485.

6.3.5.1. MAX-3485

[16] És un tipus de transmissor-receptor per a comunicacions RS-485 i RS-422 .

Aquest tipus de component ens permet transmetre informació a una velocitat màxima de 10 Mbps.

El comunicador és un petit circuit de corrent limitat i protegit contra l'excés de dissipació de potència degut a l'escalfament produït quan a les sortides s'hi crea una gran impedància.

El dispositiu presenta una característica que garanteix una sortida lògica alta, si les dues entrades són un circuit obert.

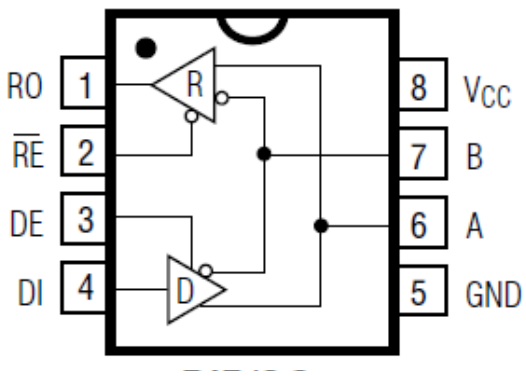
Pot ser utilitzat en:

- Control de xarxes locals i industries.
- Telecomunicacions
- En circuits de comunicació de dades.

Característiques elèctriques en DC:

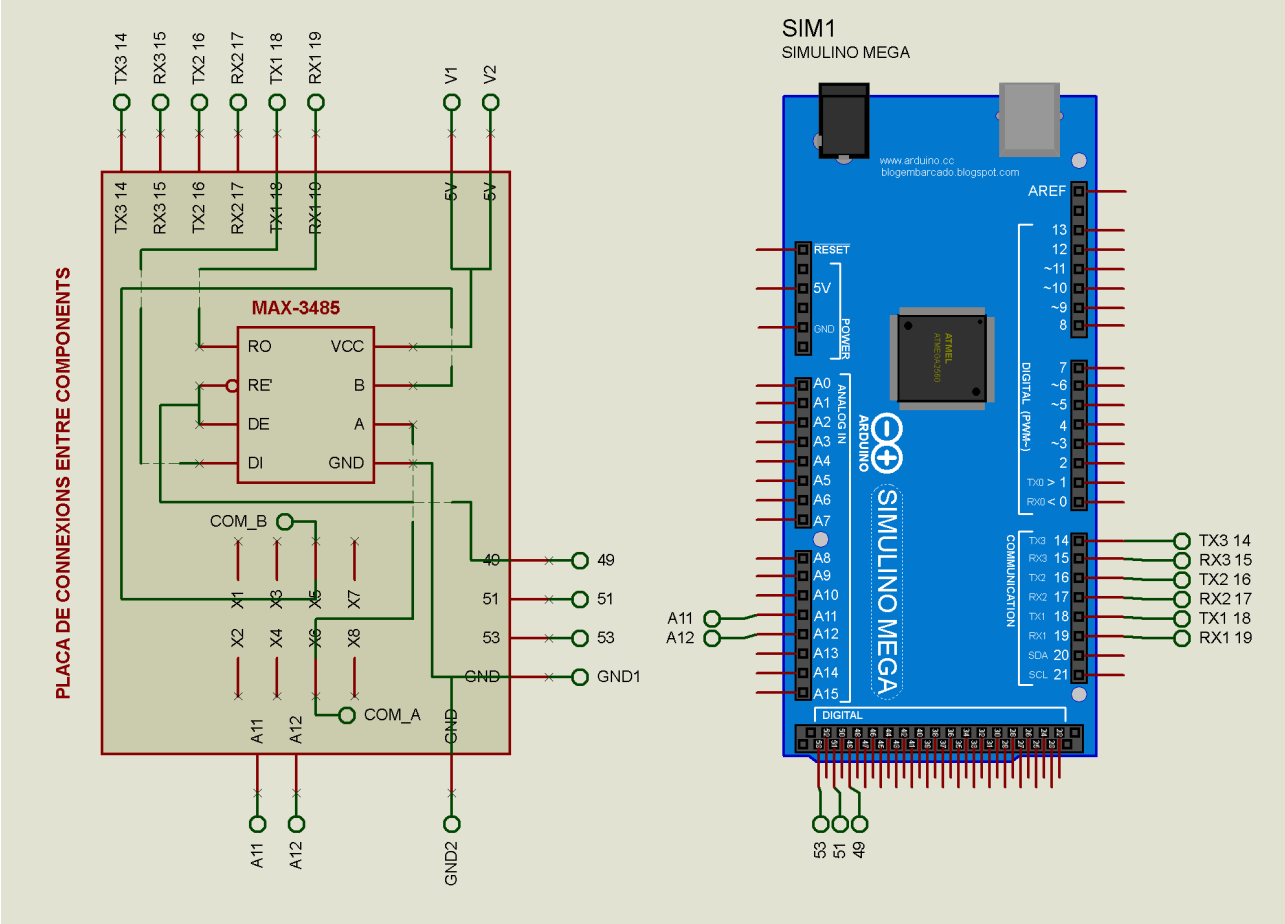
- Voltatge d'alimentació (Vcc) : 7V.
- Voltatge de control d'entrada : -0,3 fins a 7 V.
- Voltatge d'entrada de conducció : -0,3 fins a 7 V.
- Voltatge de sortida de conducció: -7,5 fins a 12,5 V.
- Voltatge d'entrada de recepció: -7,5 fins a 12,5 V.
- Voltatge de sortida de recepció: -0,3 V fins a (Vcc +0,3 V).

Il·lustració 33: MAX-3480.



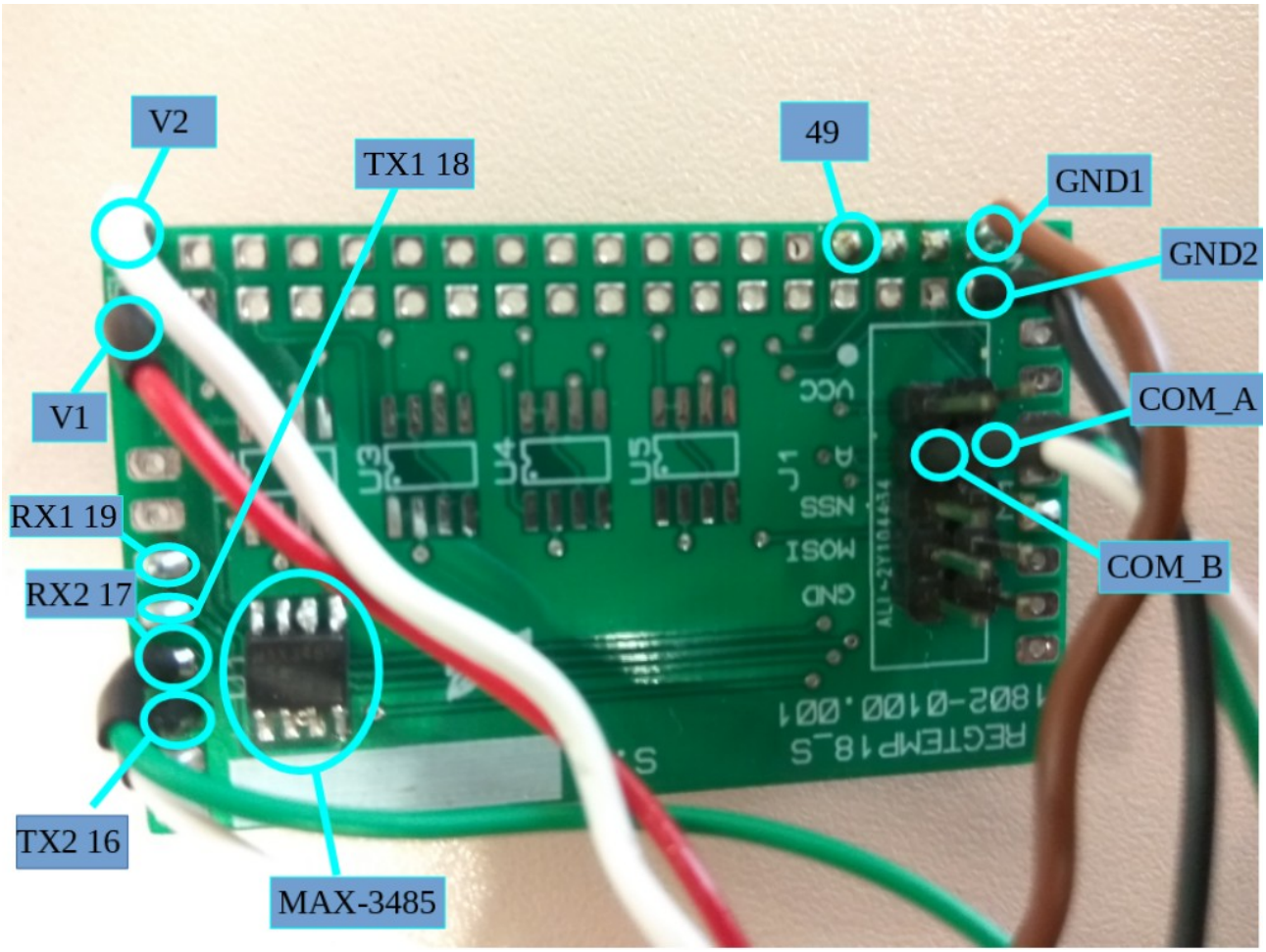
Il·lustració 34: Pin layout MAX-3480

Per explicar el circuit de la placa s' utilitzaran els noms emprats en el següent esquema realitzat amb proteus.



Il·lustració 35: Esquema simplificat de la placa de connexió amb Arduino i les seves connexions, realitzada amb Proteus.

Tal i com es pot observar amb claredat al dibuix hi han una serie de pins que no tenen nom, doncs bé, aquests són els que no són utilitzats, simplement s'han posat a la placa per a possibles modificacions futures en el disseny d'aquesta.



Il·lustració 36: Placa connexió amb Arduino, part davantera, amb les parts principals destacades.

A continuació s'explicarà exactament la funcionalitat de cada pin/component.

- El MAX-3485, com ja s'ha esmentat, ens permet transmetre i rebre informació. Per tant, aquest únicament té la funció d'enviar/rebre les dades al Swift Rail RS, quan des del microcontrolador se li doni l'ordre. És a dir, a nivell de funcionalitat aquest xip permet una definició de papers de comunicació (emissor/receptor) entre l'Arduino i el Swift Rail RS. També ens facilitat l'enviament/recepció de les dades.

- Tant el pin V1 i V2, que corresponen als cables vermell i blanc, s'encarreguen d'alimentar el dispositiu MAX-3485. A pesar de que van connectats en la connexió de 5 V del arduino, aquesta placa no realitza la corresponent alimentació. La potència es subministrada pel transformador RS-15-3,3 perquè el xip necessita ser alimentat a 3,3 V.

  Doncs dels borns de connexió V1 i V2 fins al pin numero 8 del MAX-3485 hi ha un circuit imprès en la placa PCB.

- Els pins GND1 i GND2 com la seva nomenclatura indica, són les posades a terra.

  Coincideixen amb les connexions del terra del dispositiu Arduino.

  Aquest terra està connectat amb el pin numero 5 del MAX-3485 a través del PCB.

- Els pins RX1 19 i TX1 18 a la placa van connectats a RO i DI del xip MAX-3485. Aquests són els encarregats de transmetre/rebre la informació del Swift Rail-RS. Per tant seran uns pins comuns tant en el prototip amb Sigfox i SIM card.
- Els pins RX2 17 i TX2 16 no tenen cap finalitat en la placa, únicament serveixen per establir la comunicació de Transmissió/Recepció del microcontrolador Arduino a l'element encarregat de transmetre les dades, que és el Sigfox d'acord amb el codi ja implementat per l'empresa.
- COM_A presenta un cable verd que es dirigeix a una entrada RS485 del Swift Rail-RS.

  Per tant, s'encarrega de transmetre una part de la informació del pes.

  Doncs aquesta informació prové del MAX-3485, del pin 6. Està connectat amb el pin COM_A a traves d'un circuit imprès en la PCB.
- COM_B és el cable blanc, que va dirigit directament a l'altra entrada RS485 del Swift Rail-RS.

  Juntament amb COM_A transmeten el pesatge perquè l'usuari pugi veure el resultat. Aquest informació prové del pin 7 del MAX-3485.

  La connexió entre el pin 7 i COM_B també es realitza a través d'un circuit integrat en la PCB.
- El pin 49, que aparentment únicament es veu una soldadura, presenta un circuit integrat des d'aquest, fins als pins 2 i 3 del MAX-3485, els quals estan units.

  Els pins 2 i 3 del xip són el (RE)' i el DE, que s'encarreguen d'establir el mode del xip MAX-3485. Aquesta unió si es troba en estat "high" o 1 lògic, implica que en el DE hi hagi un 1 i en el (RE)' un 0 "low" , per tant, permet una fàcil alternança de papers comunicatius entre el microcontrolador i el Swift Rail RS.

7. Processos de disseny de l'electrònica del prototip

Aquest apartat inclou tots els processos que es realitzen per dissenyar l'electrònica del prototip. Està format per una programació del microcontrolador, una breu explicació de les comandes AT i la menció de les empleades. Seguidament, es tracta també el protocol MQTT i la seva relació amb la base de dades.

7.1. Programació del microcontrolador

Pel que fa la programació del microcontrolador MEGA 2560 s'ha realitzat a través del programa Arduino, que ens permet una fàcil implementació del codi i transmissió en el xip d'aquest.

El llenguatge utilitzat és c++ juntament amb comandes AT per l'enviament de les dades de forma telemàtica.

Per superar els tallafocs que eviten l'entrada de dades exteriors a l'empresa s'ha utilitzat el protocol MQTT.

En els següents apartats s'exposa la informació relativa a les comandes AT i el protocol MQTT utilitzats.

7.1.1. Comandes AT

Les comandes AT són instruccions codificades que conformen un llenguatge de comunicació entre el programador/a i el terminal mòdem.

La principal finalitat de les comandes AT és la comunicació entre mòdems. Pel que fa a la telefonia, aquesta també utilitza com a estàndard el llenguatge per comunicar-se amb els seus terminals.

Per tant, aquestes comandes permeten a través d'una programació de baix nivell establir connexions a la xarxa a traves d'una targeta SIM, i realitzar múltiples operacions a partir d'aquesta.

Aquestes comandes retornen en el "buffer" del microxip unes respostes per a cada comanda introduïda per tant, permeten fer un seguiment continu del procès.

7.1.1.1. Comandes AT aplicades

[17] De l'immens ventall de comandes AT existents, s'han seleccionat les següents. Totes elles han d'estar aplicades amb un ordre seqüencial perquè van estretament lligades.

- AT: Serveix per comprovar si la connexió entre mòduls encarregats de realitzar la comunicació és correcta.
  - Resposta esperada: OK
- AT+CPIN=XXXX: Utilitzada en cas que la targeta SIM estigui codificada amb la seguretat PIN. Per aquest cas no s'ha utilitzat perquè prèviament he eliminat aquest pin perquè a gran escala és un inconvenient introduir tots els pins diferents per a cada cas concret.
  - Resposta esperada: +CPIN:READY OK
- AT+CSTT=APN,username,pasword : Permet establir la connexió entre l' APN, a través de l'usuari i la contrasenya, per després poder iniciar la connexió a internet.

  En aquest cas concret ha estat: AT+CSTT=(movistar.es, ,)
  - Resposta esperada: OK
- AT+CIICR inicia la connexió GPRS d'acord amb l' APN establert a la comanda anterior.
  - Resposta esperada: OK
- AT+CIFSR: retorna la IP local. Anteriorment la connexió a d'haver estat iniciada per a utilitzar aquesta comanda AT.
  - Resposta esperada: XXX.XXX.X.XX
- AT+CIPSTART=(TCP/UDP),(Domain name),(port): Ens permet establir una connexió TCP o UDP.

  En aquest cas ha estat: AT+CIPSTART=TCP, test.mosquitto.org, 1883
  - Resposta esperada: CONNECT OK
- AT+CIPSEND: Permet enviar dades.
  - Resposta esperada: >...dades a enviar....

    SEND OK
- AT+CIPSHUT: Tanca la connexió GPRS.
  - Resposta esperada: SHUT OK

7.1.2. Protocol MQTT

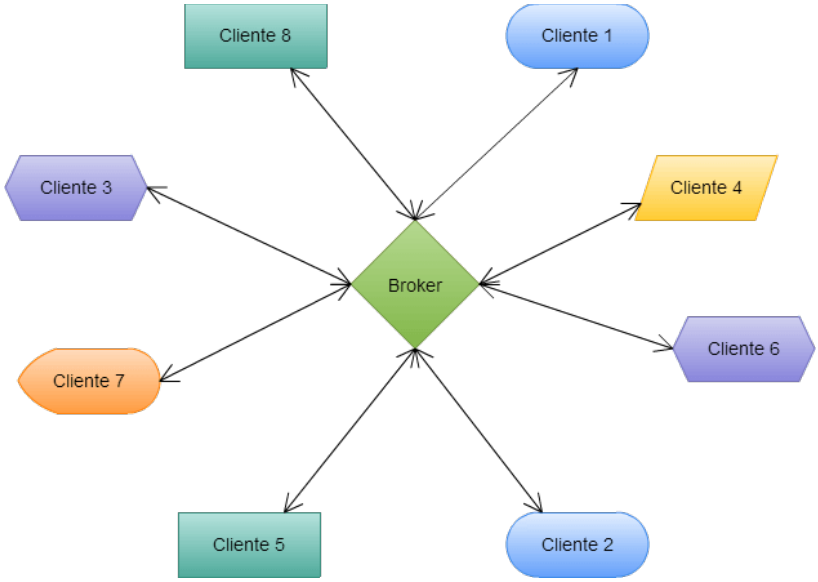
[17] És un protocol de connectivitat m2m basat en un altre protocol, el TCP/IP.

És dissenyat pel transport de dades de poc pes, a traves d'un "publish", "subscribe". Per tant, està pensat per la comunicació de sensors.

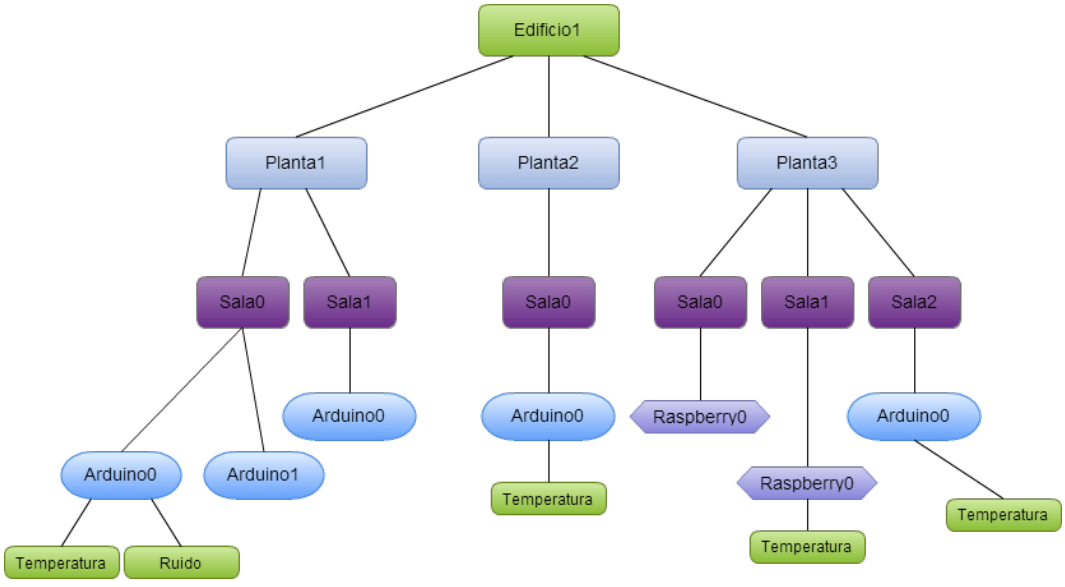
L'arquitectura de MQTT segueix una topologia d'estrella, que presenta un node central que fa de servidor o de "broker", i té una capacitat de fins a 10000 clients. El "broker" és l'encarregat de mantenir actiu el canal, i els clients hi envien periòdicament missatges. Aquesta comunicació pot ser xifrada.

La comunicació està basada amb uns "topics" que el client que publica estableix i els nodes que vulguin obtenir la informació s'han de subscriure a ell. Aquests "topics" es representen mitjançant una cadena i són de tipus jeràrquica. D'aquesta manera podem obtenir jerarquies de clients que publiquin i rebin dades de diferents branques de l'arbre jeràrquic del "topic" general.

Les següents dos figures són exemples d'estructura de nodes i jeràrquica típica del protocol MQTT.



Il·lustració 37: Estructura de nodes del protocol MQTT.



Il·lustració 38: Exemple d'estructura jeràrquica d'un "topic", en aquest cas "Edificio1".

7.1.2.1. Protocol MQTT aplicat

Aquest protocol ha estat utilitzat com a solució als tallafocs presents en la xarxa de l'empresa. Enviar una dada de forma directa a través d'un port i la IP externa de l'empresa resulta impossible amb aquest microcontrolador ja que saltar els tallafocs és una tasca complexa. Per tant, a través del protocol MQTT s'aconsegueix fer una ruta alternativa de les dades sense tenir que saltar les seguretats de la xarxa. Les dades passen temporalment a un broker, des d'on poden ser agafades posteriorment des de qualsevol ordenador.

Per tal de poder utilitzar aquest protocol per enviar dades a partir d'una targeta SIM , és a dir senser formar part d'una xarxa local, s'ha tingut d' utilitzar aquest protocol al més baix nivell. Per la comunicació no local no hi ha llibreries en C++ capaces d'interactuar amb les comandes AT i el protocol MQTT.

Doncs bé, el que s'ha esmentat com a comanda "publish" no pot ser realitzada si prèviament no s'estableix una connexió entre el "broker" i el client. Aquesta connexió és fa amb la comanda "connect".

Per tant, primerament s'ha de portar a baix nivell aquesta comanda i seguidament la "publish".

Comanda "connect"

Aquesta comanda és un vector que presenta la següent estructura:

$$[c\ p\ t][r\ len][prot\ len][prot][prot\ lvl][conf l][k\ a\ i][cl\ ID\ len][cli\ ID][U\ len][Use\ name][Pw\ len][Pw]$$

on:

- c p t = Control packet type.

  $[10]$ indicarà al broker que és un paquet de tipus connect.

- r len= remaining length.

  $[x]$ És un valor amb hexadecimal de la longitud fins al final del vector .

- prot len= protocol length.

  $[00][04]$ Perquè el protocol és MQTT, i té 4 bits de longitud la paraula.

- prot= protocol.

  $[4D][51][54][54]$ Que en decimal és MQTT.

- prot lvl= Protocol level.

[04] Nivell per defecte.

- con fl= connect flags.

  [x2] Depèn de si s'utilitza usuari i contrasenya o bé sense.

  - [02] Significa que no s'utilitza ni contrasenya ni usuari.
  - [C2] Significa que sí s'utilitza usuari i contrasenya.

- k a i = keep alive intervals.

  [00][3C] Amb decimal és 60, que equivalen a segons.

- cl ID len= client ID length.

  [x][x] Depèn del ID que es posi.

- cli ID= client ID.

  [x]....[x] Nom amb hexadecimal del ID client.

- U len= User length.

  [x][x] Valor amb decimal de la longitud del nom d'usuari.

- Use name= User name.

  [x][x] Valors amb hexadecimal del nom d'usuari de tipus ASCII.

- Pw len= Password length.

  [x][x] Valor amb decimal de la longitud de la contrasenya.

- Pw = Password.

  [x]....[x] Nom amb hexadecimal de la contrasenya de tipus ASCII.

En el cas concret del prototip dissenyat no hi ha ni usuari ni contrasenya, per tant els quatre últims components del vector no hi seran presents.

Pel que fa al Client ID que s'utilitza serà "ABCDEF" que en hexadecimal és "414243444546".

Per tant, el vector connect a introduir al microxip és:

$$connect = [10][12][00][04][4D][51][54][54][04][02][00][3C][00][06][41][42][43][44][45][46]$$

Comanda "publish"

Aquesta comanda és un vector que presenta la següent estructura.

$$[c\ p\ t][r\ len][topic\ len][topic][mes]$$

On:

- c p t = Control packet type.

  $[30]$ indicarà al broker que és un paquet de tipus publish.

- r len= remaining length.

  $[x]$ És un valor amb decimal de la longitud fins al final del vector .

- topic len= topic length.

  $[x][x]$ És el valor decimal de la longitud del "topic".

- Topic

  $[x]....[x]$ El "topic" escrit en hexadecimal, de la longitud establerta anteriorment de tipus character.

- Mes= measurement.

  $[x]....[x]$ És el valor/text de tipus ASCII que s'enviarà a la base de dades al "topic" corresponent.

En aquest prototip el vector "publish" tindrà les següents característiques:

- Topic: "pinsos_guissona", que en hexadecimal és:

  $$[70][69][6E][73][6F][73][5F][67][75][69][73][73][6F][6E][61]$$

- I en el context de mesuraments sempre s'hi afegirà davant: "{"granja": ", que en hexadecimal és:

  $$[7B][22][67][72][61][6E][6A][61][22][3A]$$

- Seguidament s'afegirà el nom de la granja, que com a molt pot contenir 25 caràcters i cap espai.

  $$[xx]...[xx]$$

- Després del nom, una coma, que en hexadecimal és:

  $$[2C]$$

- Després s'afegeix « "pes":». En hexadecimal és:

  $$[22][70][65][73][22][3A]$$

- A continuació, el valor de pes.

  $$[yy]...[yy]$$

- Seguidament, hi ha ","status":". En hexadecimal és:

$$[2C][22][73][74][61][74][75][73][22][3A]$$

- A continuació s'hi posa l'estat de la mesura.

$$[zz]$$

- Finalment es tanca el format amb una corxeta "}". En hexadecimal és:

$$[7D]$$

Per tant, el vector final és:

$$publish=[30][xx][00][0F][70][69][6E][73][6F][73][5F][67][75][69]...$$

$$...[73][73][6F][6E][61][7B][22][67][72][61][6E][6A][61][22][3A][xx]...$$

$$...[xx][2C][22][70][65][73][22][3A][yy]...[yy][2C][22][73][74][61][74][75]...$$

$$...[73][22][3A][zz][7D]$$

En la següent figura és mostra una taula amb les equivalencies entre diferents sistemes de numeració. Ha estat seguida per l'elaboració dels vectors anteriors.

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Il·lustració 39: Taula de les equivalències entre sistema decimal, hexadecimal, octal i ASCII.

7.1.3. Base de dades.

Per poder penjar les dades obtingudes amb cada mesura, ha estat necessari la cerca d'un servidor que actuï com a "broker". És a dir hi ha d'haver un intermediari entre l'empresa i el món exterior.

Això és degut a que enviar informació externa a una IP privada és molt complicat, a causa de la gran quantitat de tallafocs que l'empresa presenta.

Dit això, l'empresa ja ha proposat a nivell superiors, la implementació d'una base de dades amb un "broker" extern, amb la pertinent seguretat amb usuari i contrasenya. Actualment, encara no està disponible, per tant per aquest treball s'ha utilitzat una base de dades pública del mateix tipus que la prevista d'implementació. L'única diferència serà que presentarà un usuari privat.

Ha estat utilitzat el "Test.mosquitto.org" i com a tòpic a subscriure's: pinsos_guissona.

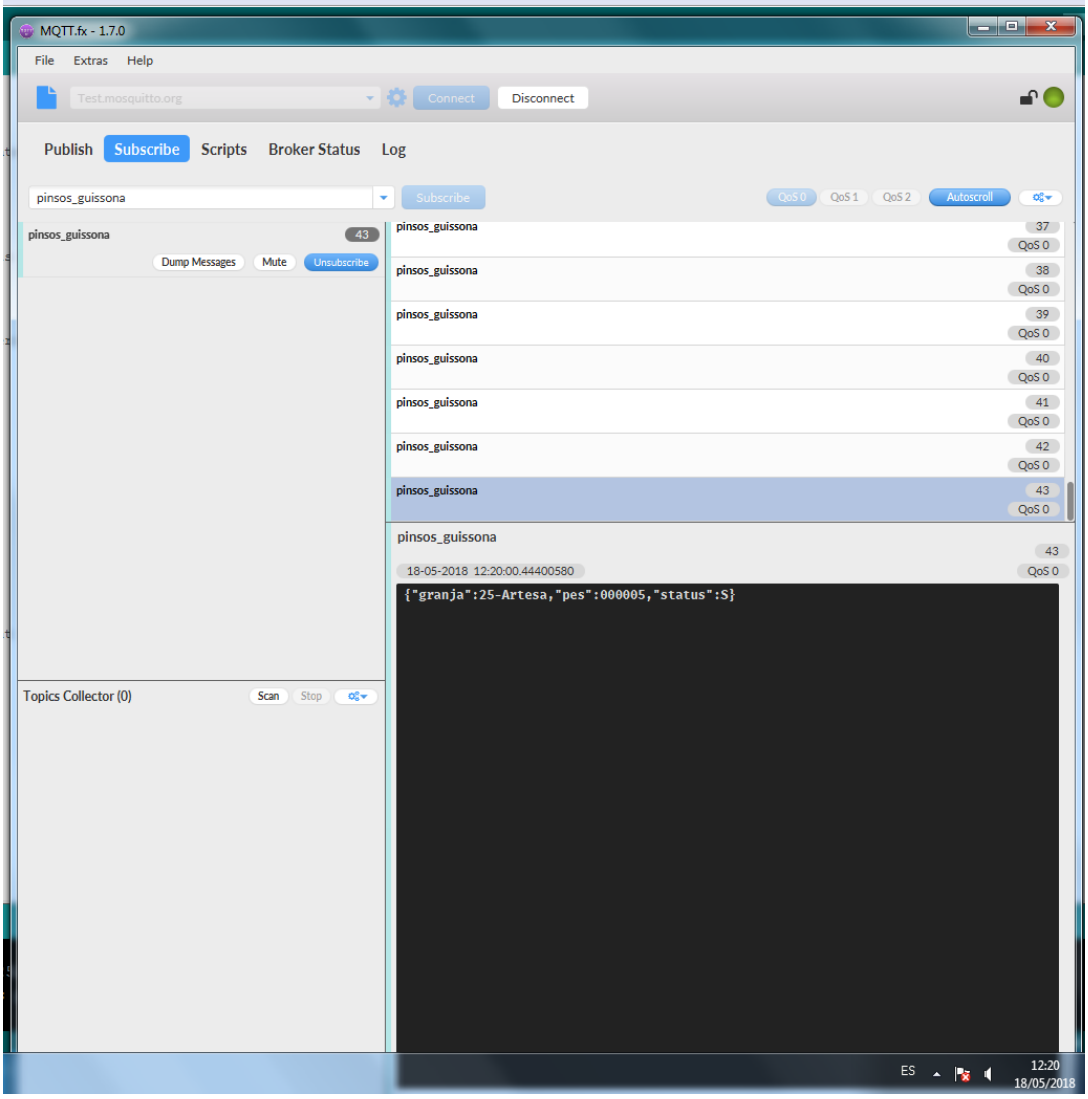
Les dades s'han enviat utilitzant un format estàndard d'estructura anomenat "Json". Té la següent forma:

{"granja": nom granja, "pes": mesura, "status": estat de la mesura}

D'aquesta forma s'estableix una identificació i una estructura de cada dada i el processat que es farà serà més fàcil.

També s'hi incorpora l'estat, que en un apartat posterior és explicat.

La següent imatge és una captura de pantalla del servidor rebent dades.



Il·lustració 40: Captura de pantalla de la base de dades, amb un exemple amb 43 dades introduïdes.

7.1.4. Obtenció de dades a través del bus de comunicació RS-485

El Swift Rail-RS permet una transmissió de les dades que ell mateix obté a través del bus de comunicació RS-485 entre altres.

Per fer aquesta comunicació primerament s'ha de definir un "mestre" i un "esclau".

Es coneix com a mestre aquell dispositiu que envia dades, i com a esclau aquell que les rep.

Per fer aquest nomenament de papers es necessita incorporar en el microxip Mega 2560 el xip MAX-3485. Aquest xip ens permet alternar si volem enviar o rebre dades des del microcontrolador a partir de la senyal que arriba en els Pins 2 i 3 que corresponen a (RE)' i DE.

Quan (RE)' té un 1 lògic i DE un altre 1 lògic, A (pin 6) i B(pin 7) passen a ser una sortida, que transmeten les dades que arriben per DI(pin 4).

Mentre que si (RE)' és 0 lògic i DE un 0 lògic, A(pin 6) i B(pin 7) passen a ser una entrada que agafen les dades a través de RO (pin 1).

Com és evident, els pins A i B van connectats al Data + i Data – del Swift Rail-RS i els pins DE i (RE)' van al microcontrolador.

Doncs bé primerament, per obtenir les dades des del Swift Rail-RS és necessita posar el microcontrolador com a mestre i el Swift Rail-RS com a esclau. Això és degut a què el Swift no envia les dades sinó rep prèviament una sèrie de comandes en hexadecimal, però de tipus ASCII.

Aquestes comandes varien en funció de la informació que volem obtenir. En aquest cas únicament és necessita utilitzar el conjunt d'informació que conté: El pes net, brut i el pic actual.

- Primerament, necessita saber l'adreça que és vol llegir.

  En aquest cas, es llegeix de l'adreça 1. Aquesta és la direcció de comunicació sèrie del instrument; és el caràcter ASCII que s'obté sumant 80 hexadecimal amb aquesta. Per tant, l'adreça a enviar serà: $0x81$.

- Seguidament, s'ha d'enviar el caràcter "N". Per tant, el seu equivalent en hexadecimal és el "E4". S'enviarà: $0x4E$.

- Finalment s'ha de tancar la comunicació, que es fa amb la comanda de characters EOT, que en hexadecimal ja té el seu equivalent. S'enviarà: $0x04$.

Un cop realitzada aquesta part del procès s'ha de canviar l'estat dels components comunicats.

Ara el microxip Mega 2560 ha de passar a ser esclau i el Swift Rail RS mestre. Això és fa posant els pins DE i (RE)' del MAX-3485 en estat baix.

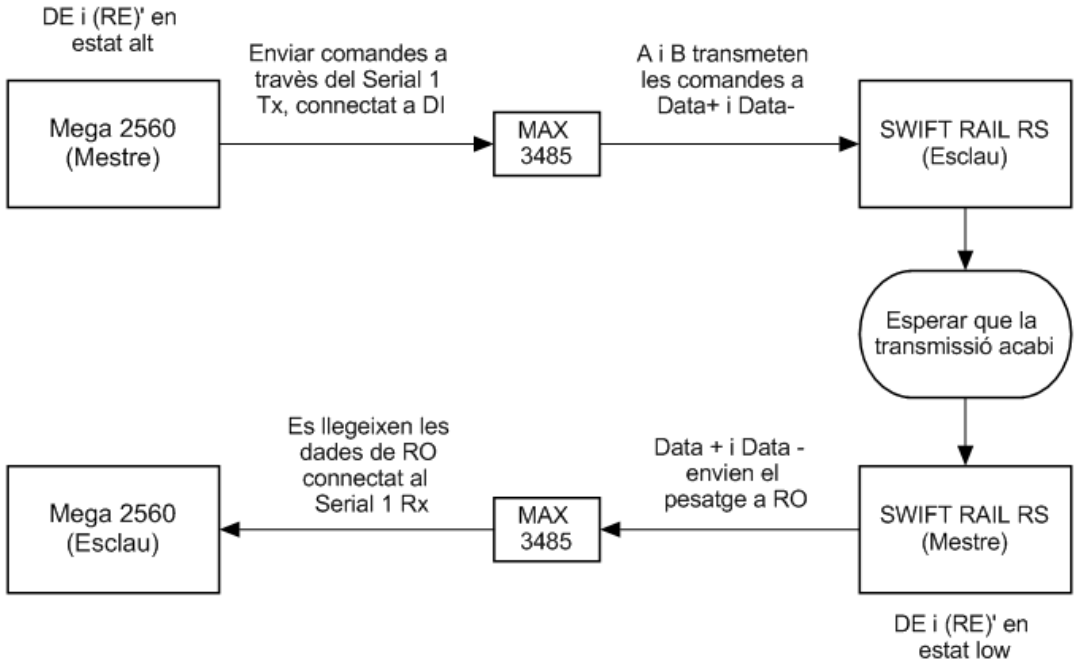
Un cop els estats definits novament, les dades del Swift Rail RS ja entren en el microcontrolador a través del serial 1 en forma d'un vector amb la següent estructura:

$$N[Adrr][status][net][gross][peak]ETX[chksum]EOT$$

On:

- Adrr: retorna l'adreça d'on és la lectura.
- Status: retorna un caràcter que accepta els següents valors:
  - S: pes estable.
  - M: Pes que no és estable.
  - O: Pes major que la capacitat màxima.
  - E: Pes que no es pot detectar.
- Net: camp que inclou el pes net en 6 caràcters.
- Gross: inclou el pes brut amb 6 caràcters.
- Peak: inclou el pes de pic amb 6 caràcters també.
- ETX: significa final del text en tipus ASCII, el seu equivalent en hexadecimal es 0x03.
- Chksum: és la codificació ASCI dels dígits hexadecimals.
- EOT: significa final de la transmissió. L'equivalent en hexadecimal és 0x04.

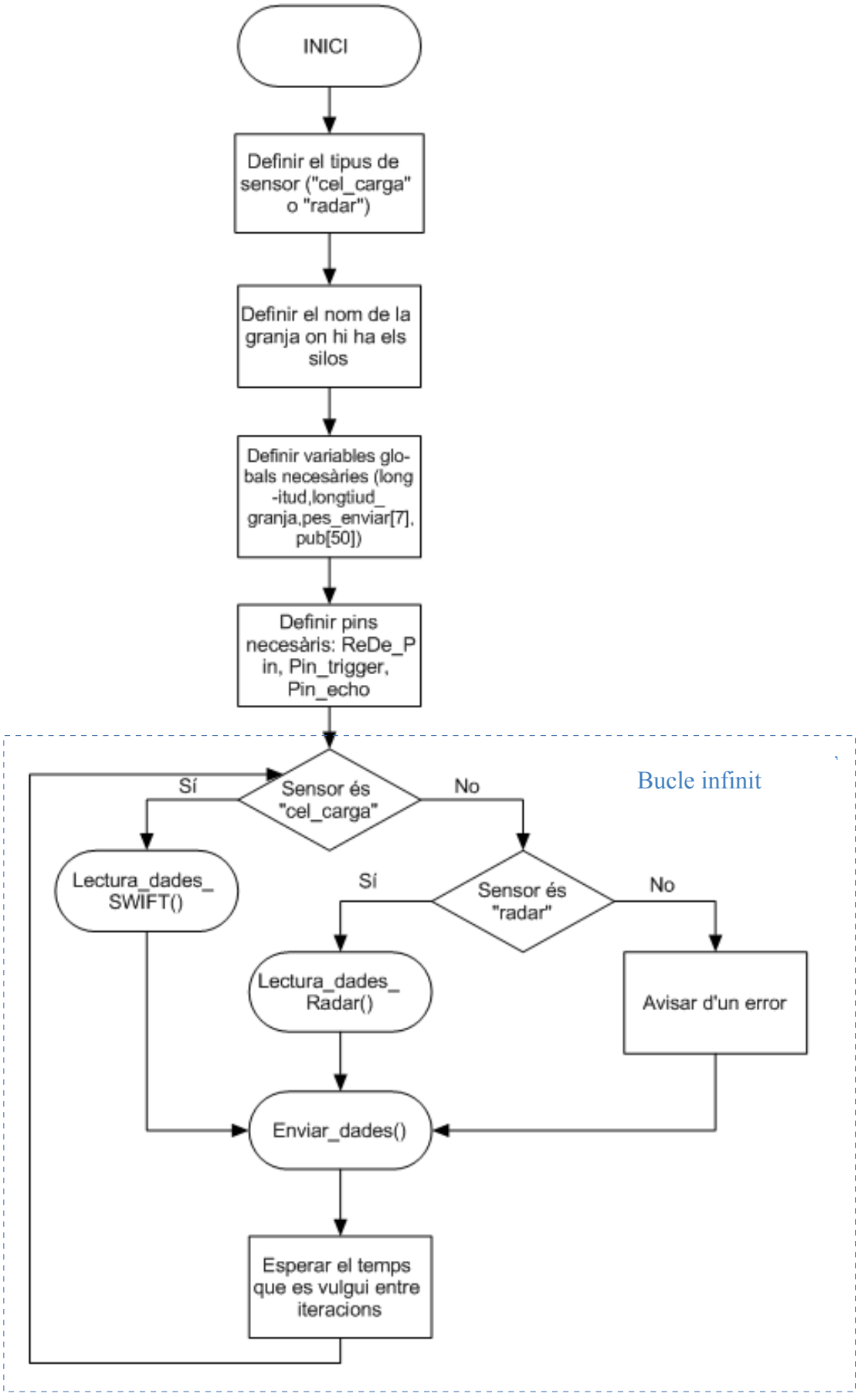
Doncs un cop és té aquest vector de sortida, únicament s'ha de tractar i agafar els components que són interessants pel cas concret. Per tant, únicament se seleccionen els 6 components que formen el pes net i el component que conté l'estat de la mesura.



Il·lustració 41: Esquema de la transmissió de dades entre el Mega2560 i el Swift Rail RS.

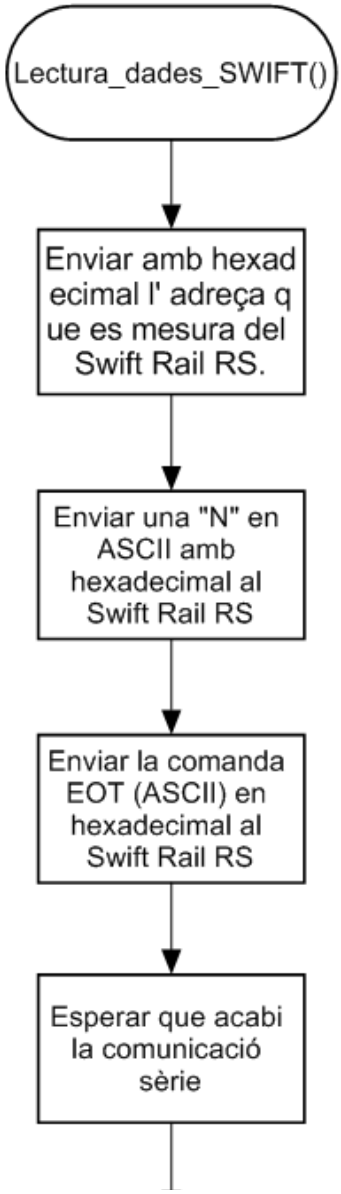
7.1.5. Diagrama de flux del prototip amb SIM Card

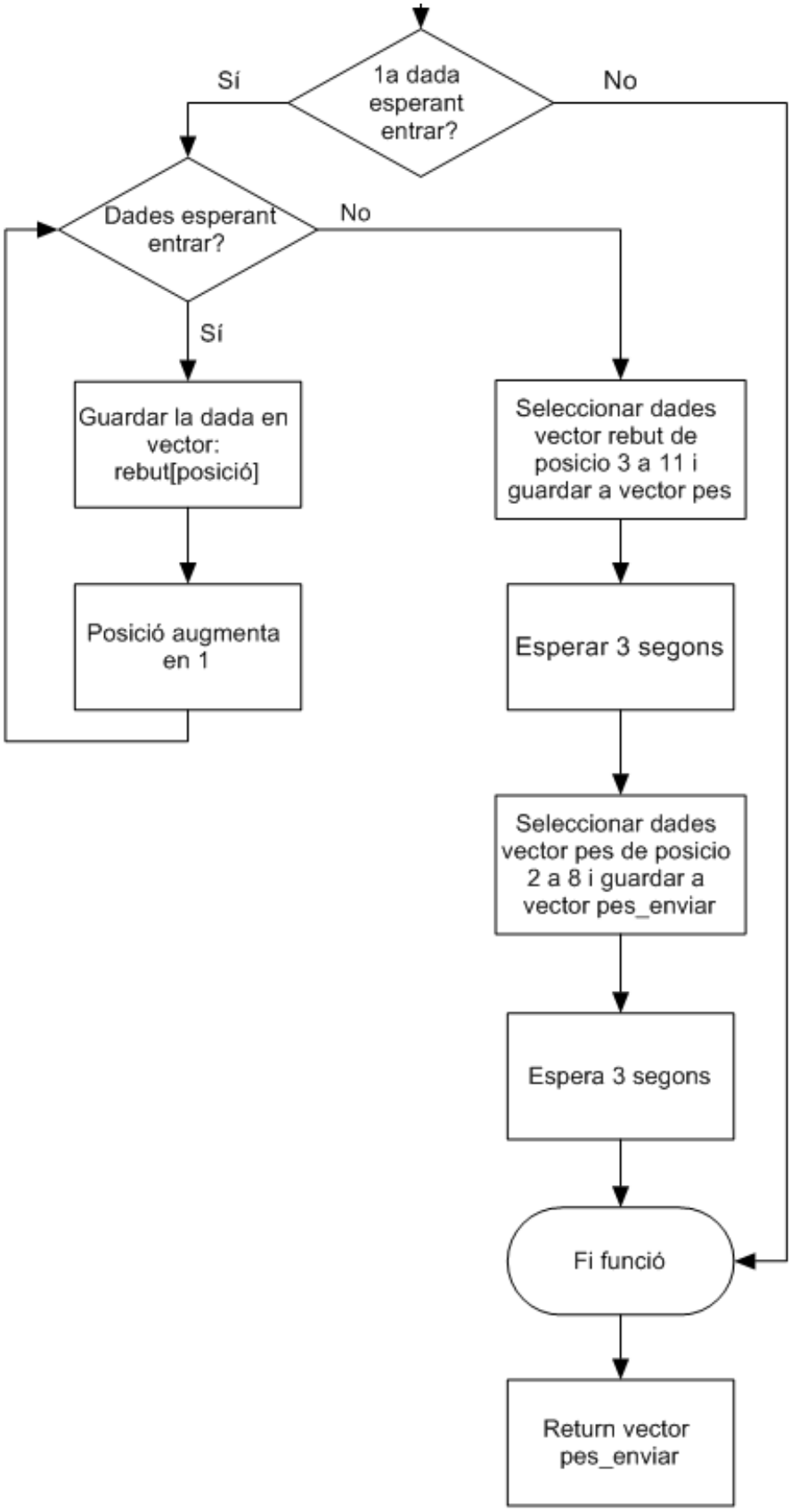
Aquest diagrama de flux és el principal del prototip elaborat. Presenta una seqüencia de processos de definició de variables/paràmetres imprescindibles que únicament s'executa una vegada, quan s'inicia el microcontrolador per primera vegada. Seguidament, hi ha un bucle infinit que depenent del tipus de sensor, executa una funció o una altra. Finalment, sempre espera el temps que es vulgui i torna a iniciar el bucle.[annex-14.1]


INICI
Definir el tipus de sensor ("cel_carga" o "radar")
Definir el nom de la granja on hi ha els silos
Definir variables globals necesàries (longitud,longtiud_granja,pes_enviar[7], pub[50])
Definir pins necesàris: ReDe_Pin, Pin_trigger, Pin_echo
Bucle infinit
Sensor és "cel_carga"
Sí
No
Lectura_dades_SWIFT()
Sensor és "radar"
Sí
No
Lectura_dades_Radar()
Avisar d'un error
Enviar_dades()
Esperar el temps que es vulgui entre iteracions

7.1.5.1. Diagrama de flux de la funció Lectura_dades_SWIFT()

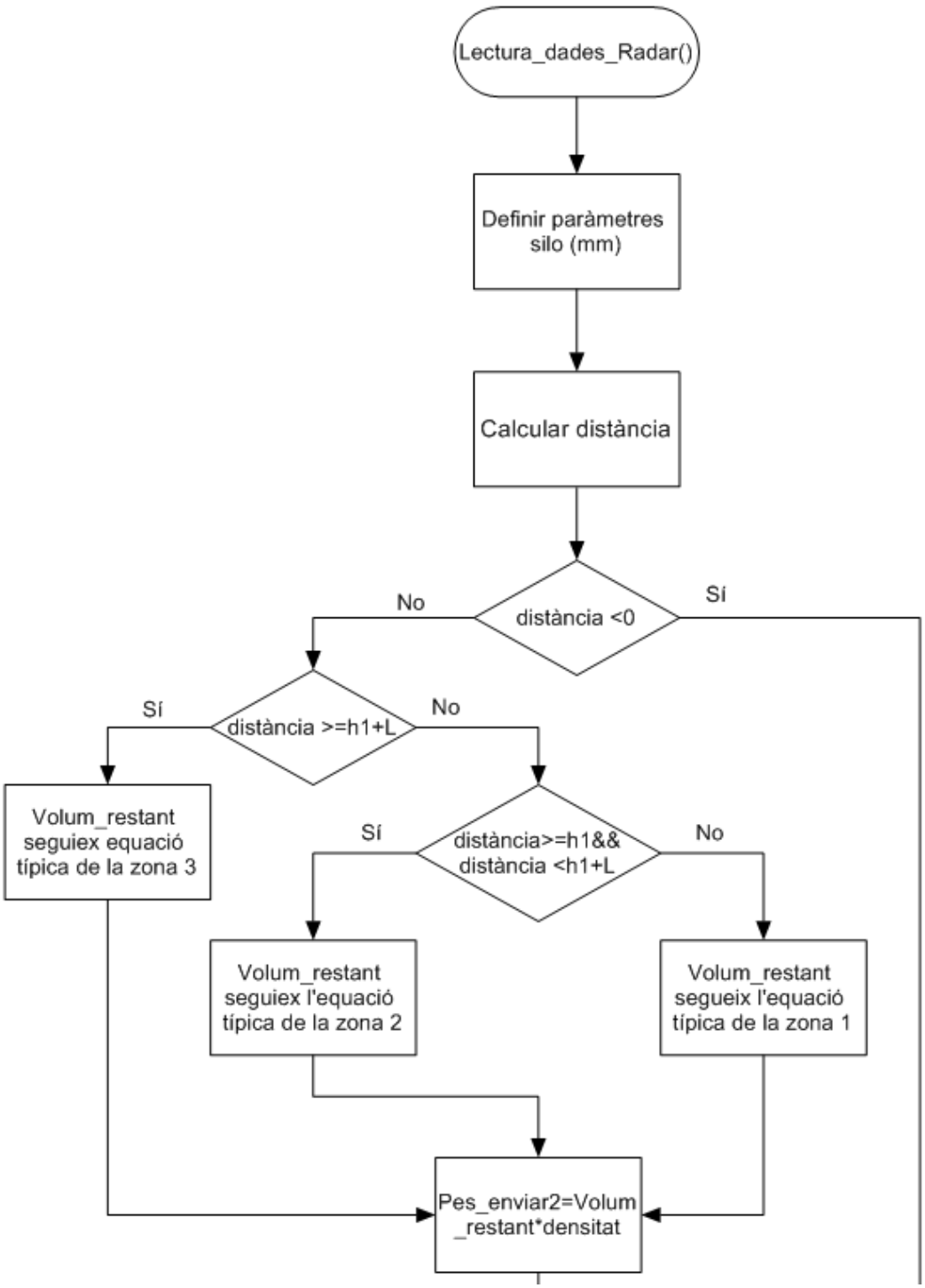
Aquesta funció s'encarrega de llegir les dades que ens proporciona el Swift Rail RS. Ho fa mitjançant una comunicació RS-485. Per tant, primerament s'estableix una comunicació entre microcontrolador i Swift Rail RS. Seguidament es guarda el valor de la lectura en el vector pes_enviar().

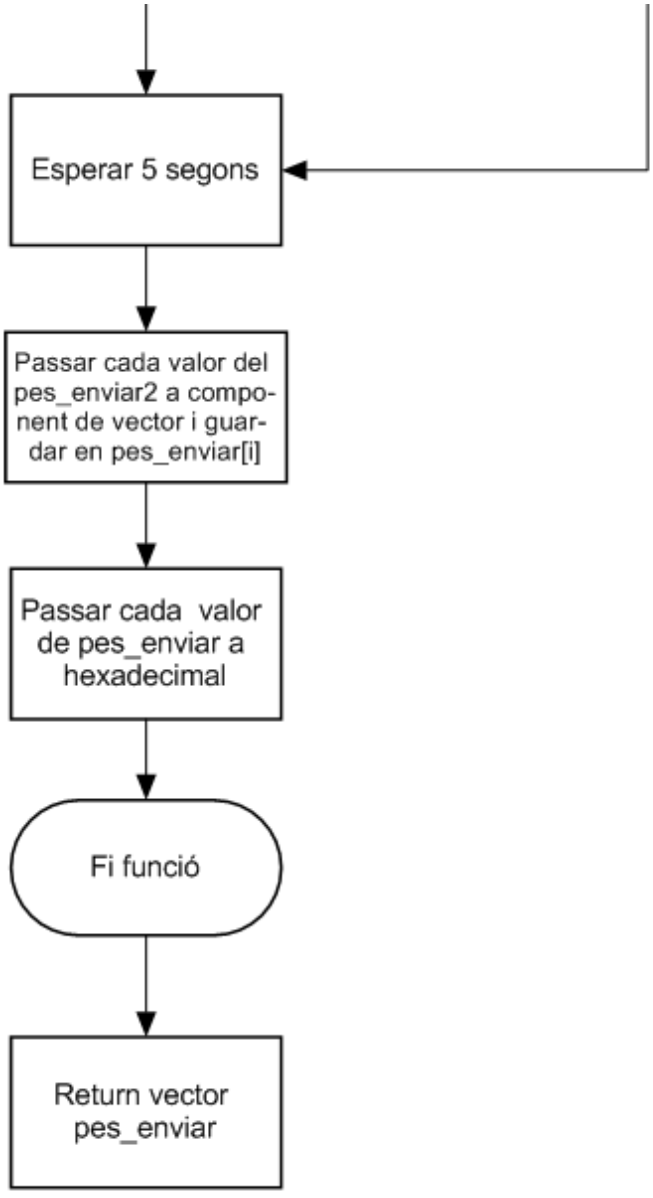



1a dada esperant entrar?
Sí
No
Dades esperant entrar?
No
Sí
Guardar la dada en vector: rebut[posició]
Posició augmenta en 1
Seleccionar dades vector rebut de posicio 3 a 11 i guardar a vector pes
Esperar 3 segons
Seleccionar dades vector pes de posicio 2 a 8 i guardar a vector pes_enviar
Espera 3 segons
Fi funció
Return vector pes_enviar

7.1.5.2. Diagrama de flux de la funció Lectura_dades_Radar()

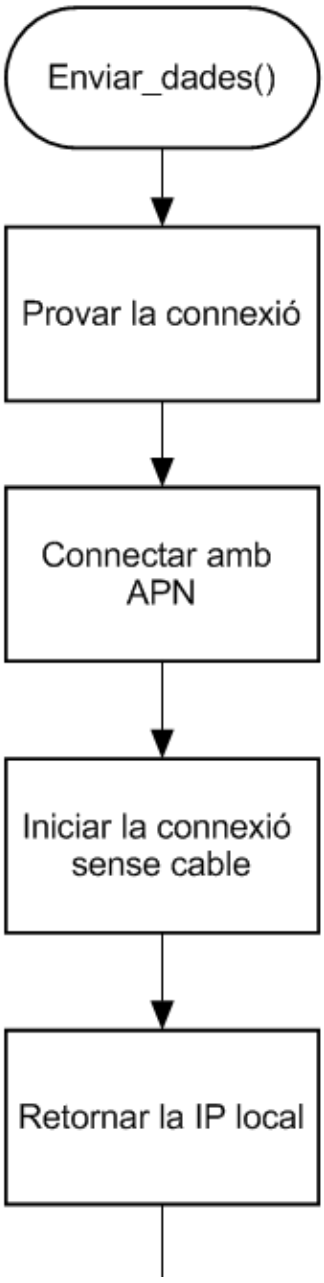
Aquesta funció llegeix la distància que el sensor radar/ultrasònic mesura i d'acord amb els paràmetres característics de la sitja i les equacions matemàtiques, calcula la massa de pinso restant. Aquest valor el transforma amb hexadecimal per tal de poder ser processat a posterior-hi.

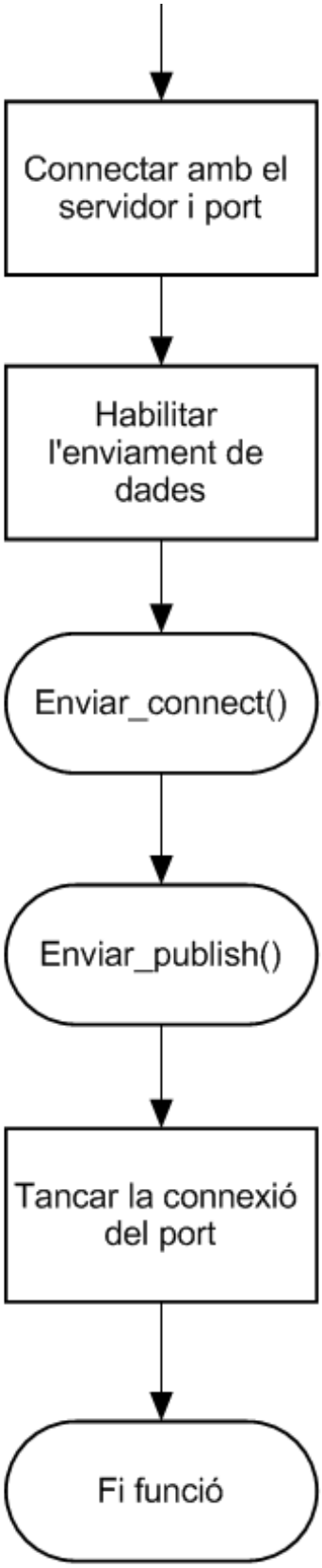



Esperar 5 segons
Passar cada valor del pes_enviar2 a component de vector i guardar en pes_enviar[i]
Passar cada valor de pes_enviar a hexadecimal
Fi funció
Return vector pes_enviar

7.1.5.3. Diagrama de flux de la funció Enviar_dades()

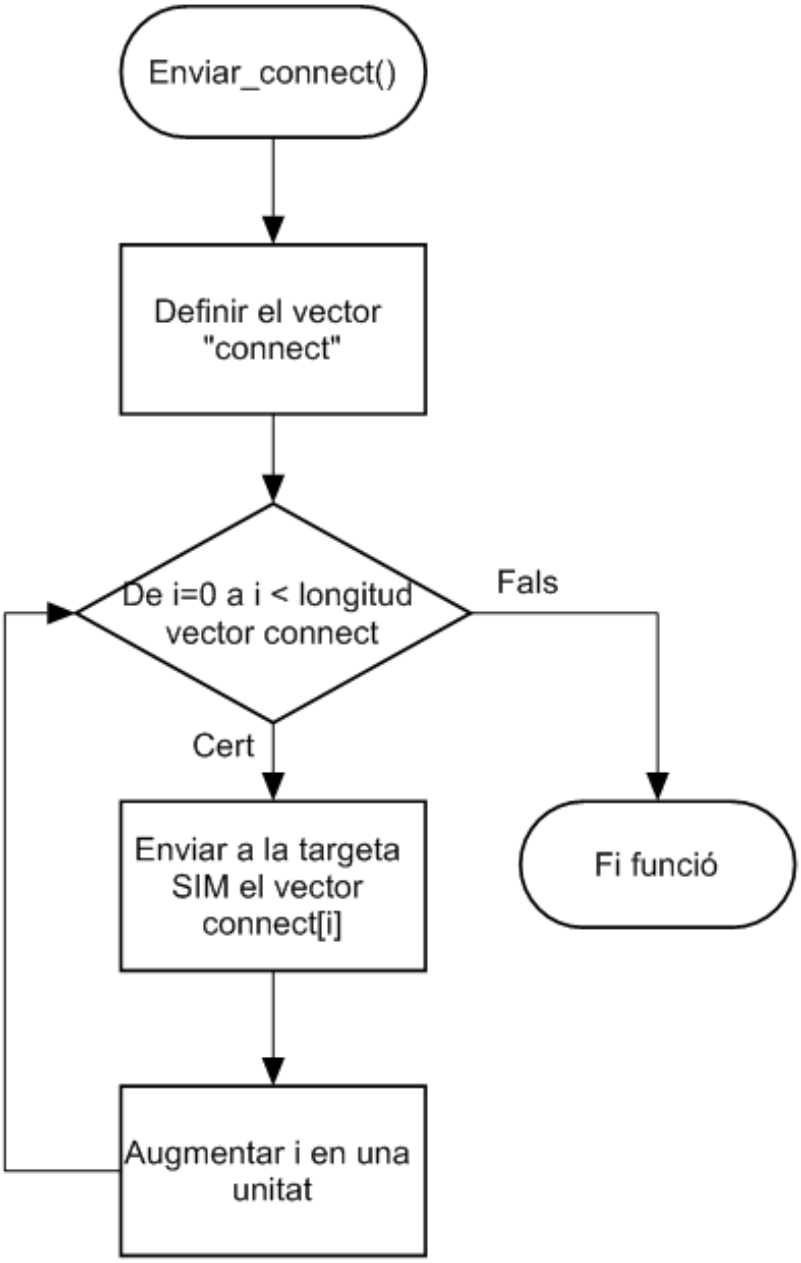
Aquesta funció s'encarrega d'enviar les dades rebudes anteriorment i guardades al vector pes_enviar(). Primerament s'estableix una connexió i es defineix el servidor i port amb el qual ens volem connectar. Un cop realitzat s'envien les dades desitjades a partir de l'execució d'altres funcions.




Connectar amb el servidor i port
Habilitar l'enviament de dades
Enviar_connect()
Enviar_publish()
Tancar la connexió del port
Fi funció

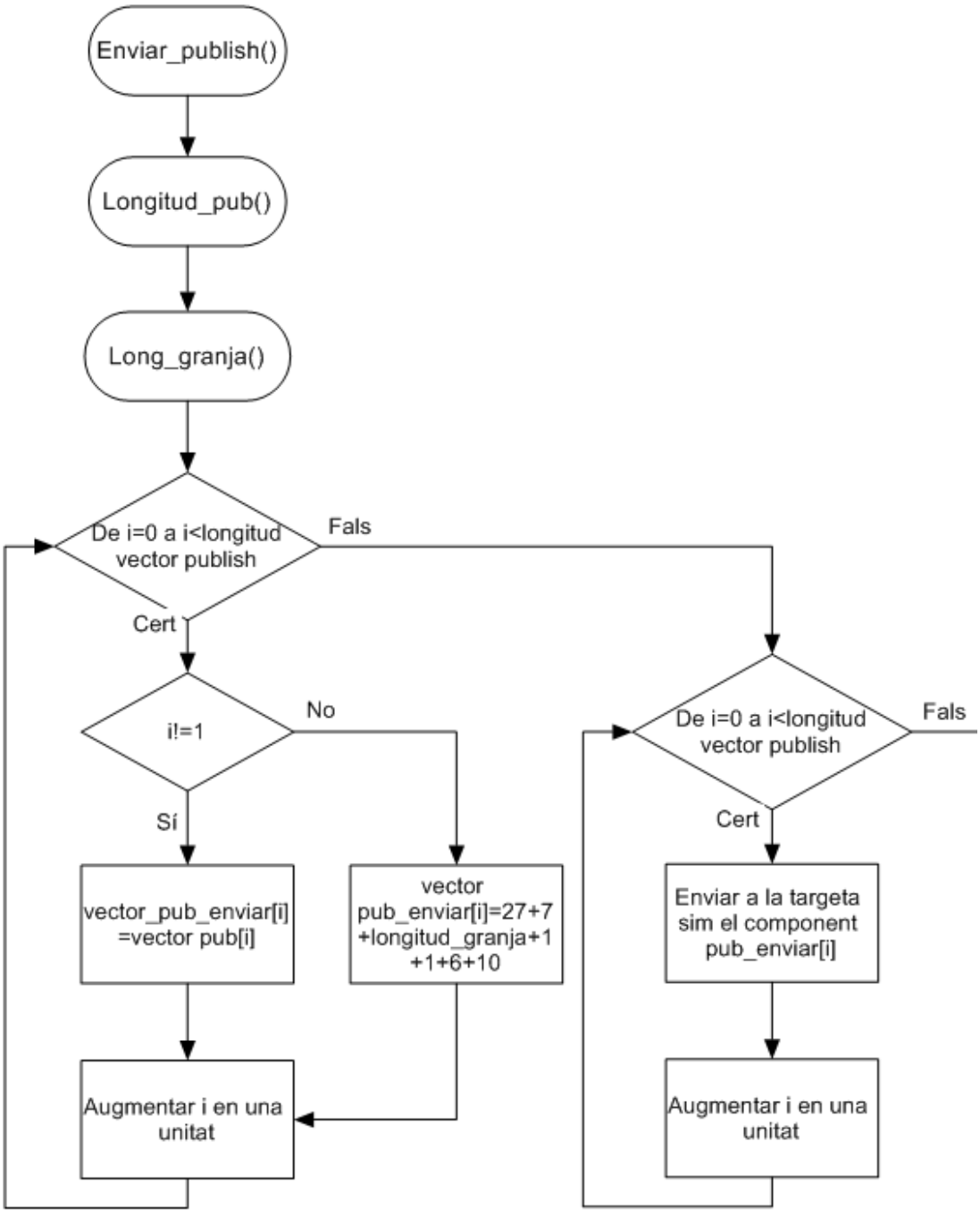
7.1.5.4. Diagrama de flux de la funció Enviar_connect()

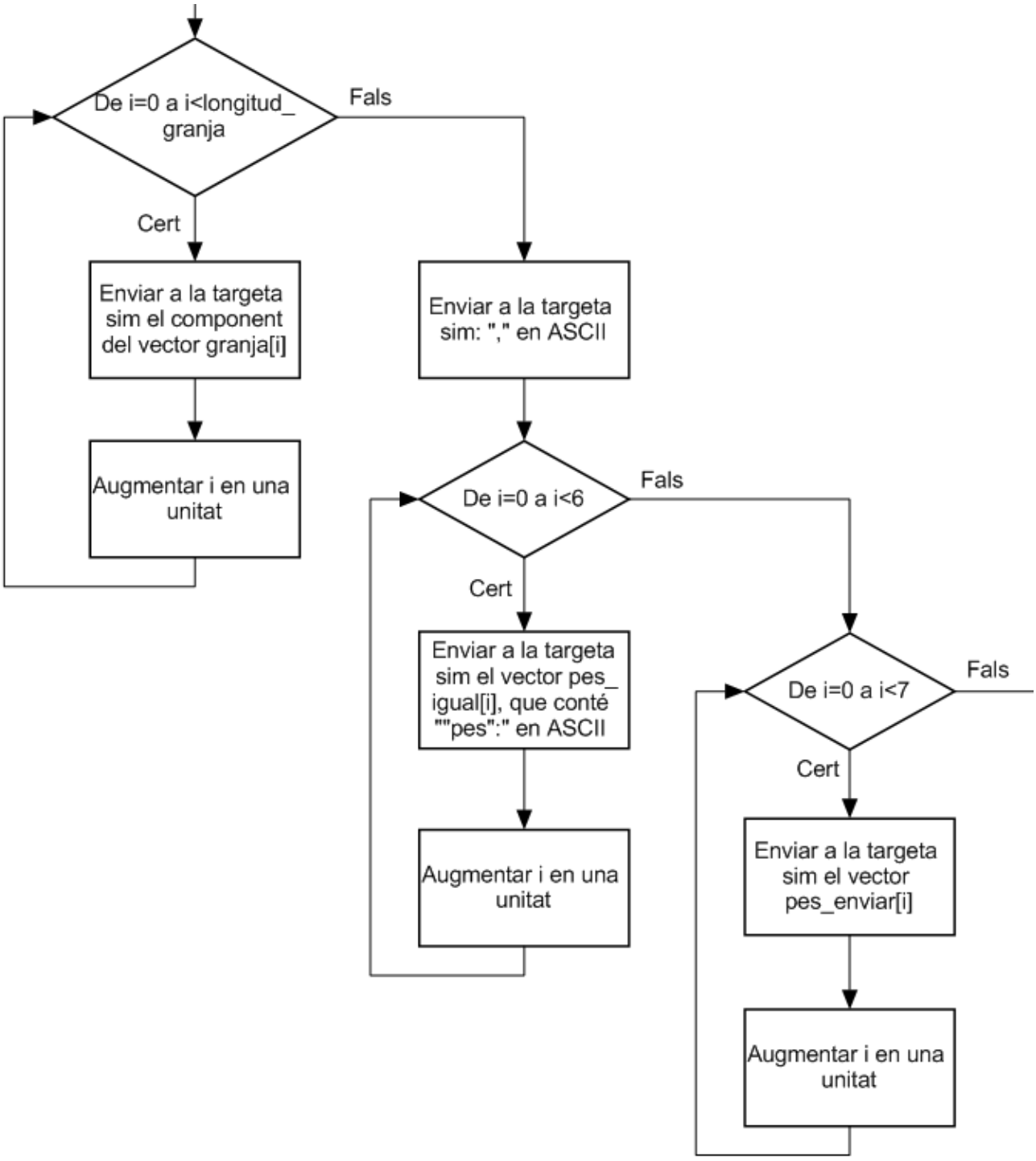
Aquesta funció s'encarrega d'enviar el vector «connect» necessari amb el protocol MQTT. Aquest vector s'envia component a component al mòdul de la targeta SIM.

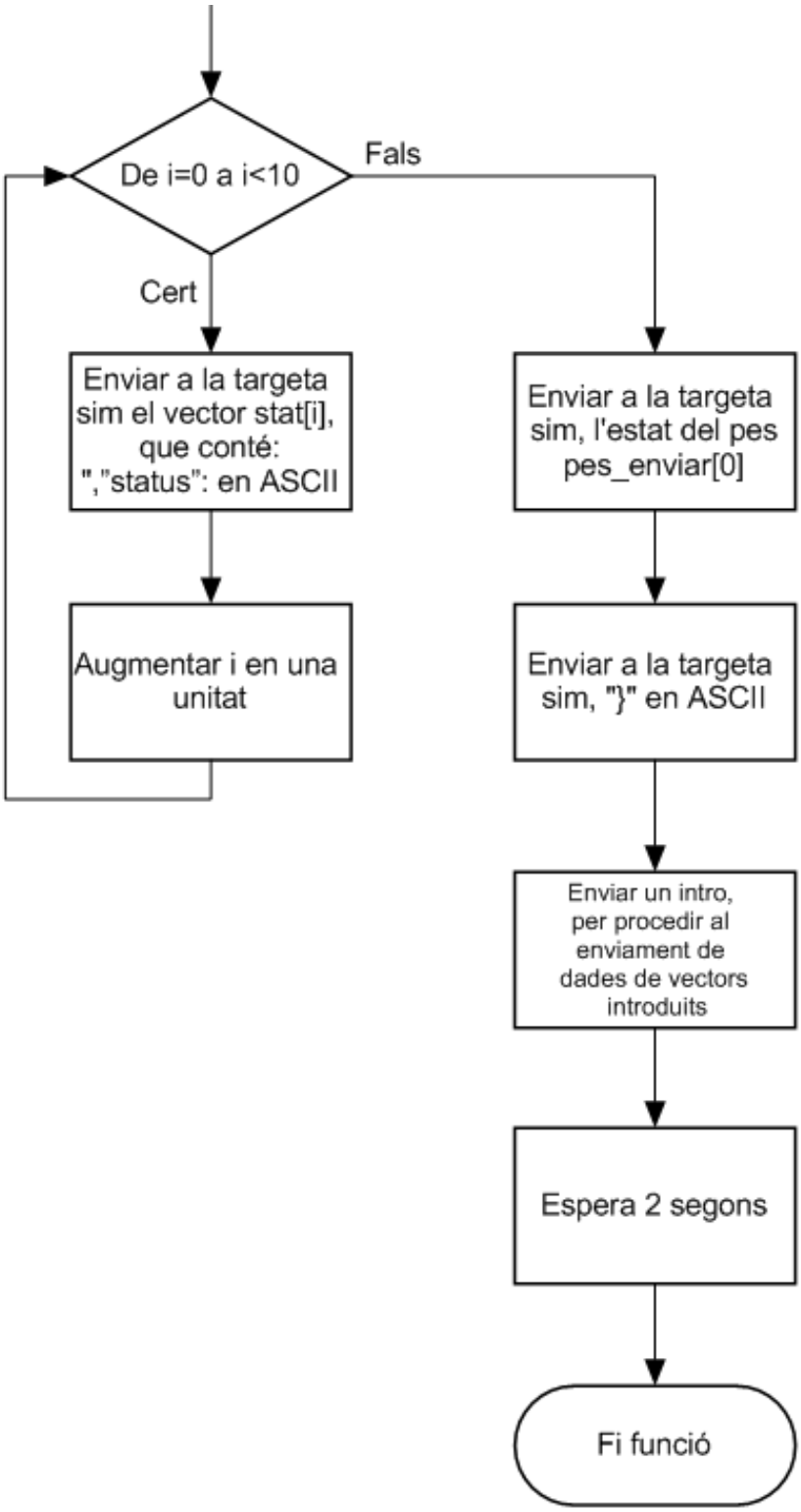


7.1.5.5. Diagrama de flux de la funció Enviar_publish()

La funció s'encarrega d'enviar el vector «publish» característic del protocol MQTT. Totes les paraules necessàries pel format Json i les dades són processades en format ASCII i enviades al mòdul que conté la targeta SIM.

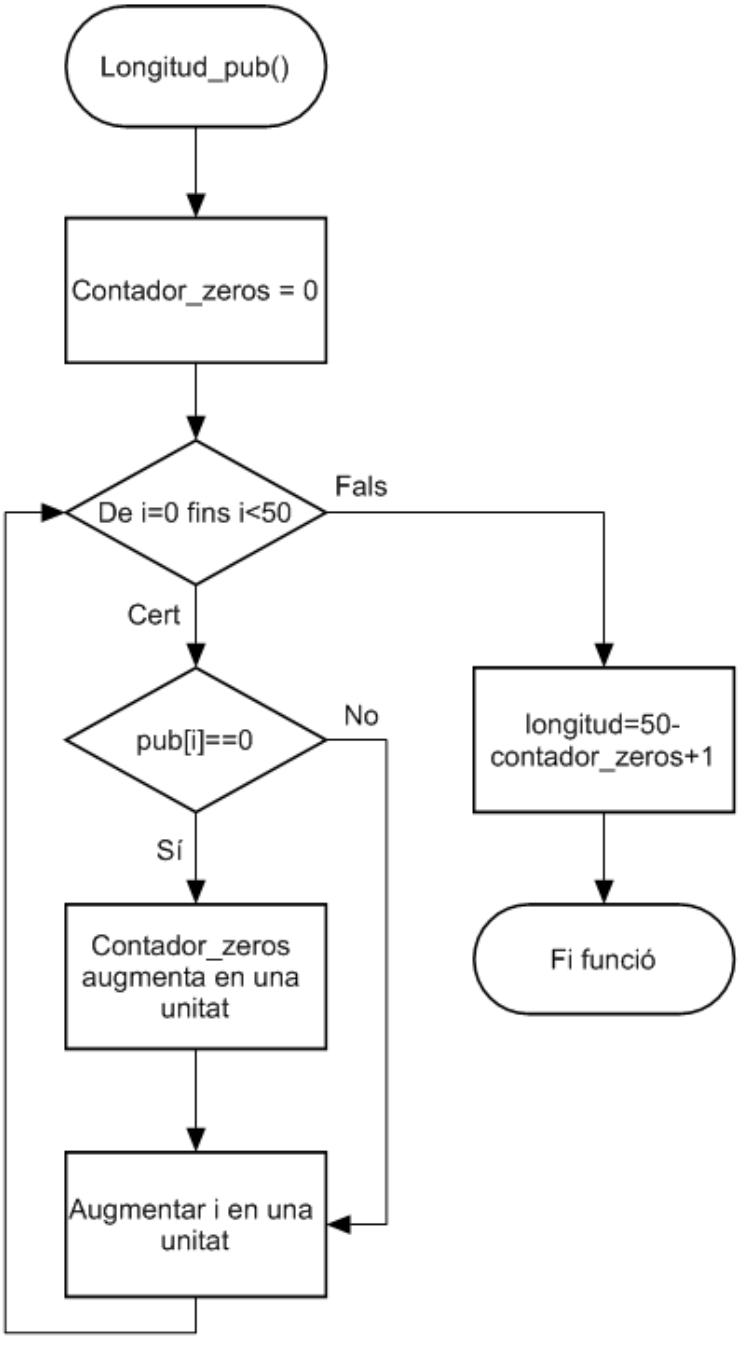



De i=0 a i<longitud_ granja
Fals
Cert
Enviar a la targeta sim el component del vector granja[i]
Augmentar i en una unitat
Enviar a la targeta sim: "," en ASCII
De i=0 a i<6
Fals
Cert
Enviar a la targeta sim el vector pes_ igual[i], que conté ""pes":" en ASCII
Augmentar i en una unitat
De i=0 a i<7
Fals
Cert
Enviar a la targeta sim el vector pes_enviar[i]
Augmentar i en una unitat


De i=0 a i<10
Fals
Cert
Enviar a la targeta sim el vector stat[i], que conté: ","status": en ASCII
Augmentar i en una unitat
Enviar a la targeta sim, l'estat del pes pes_enviar[0]
Enviar a la targeta sim, "}" en ASCII
Enviar un intro, per procedir al enviament de dades de vectors introduits
Espera 2 segons
Fi funció

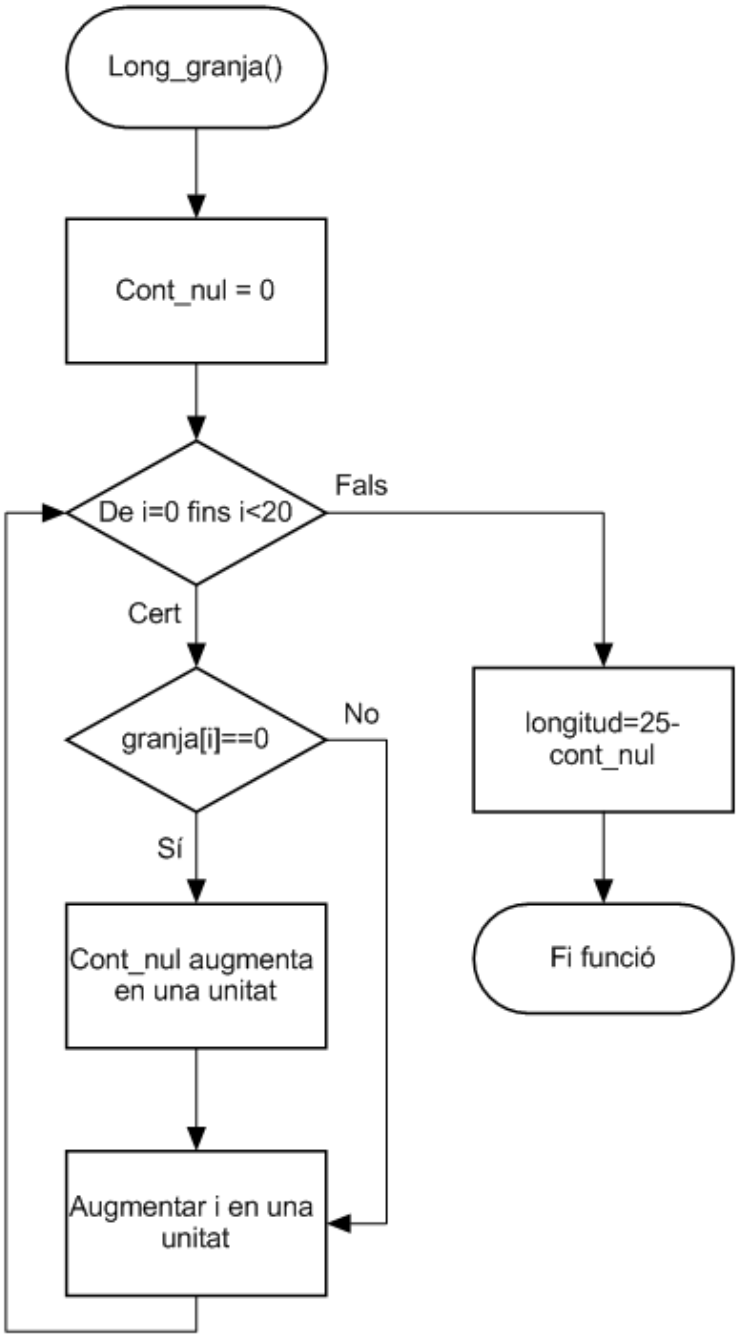
7.1.5.6. Diagrama de flux de la funció Longitud_pub()

Aquesta funció s'encarrega de mesurar la longitud del vector «publish». Recorre el vector en busca de 0 i els compta. Finalment la longitud final es el valor inicial de components menys els zeros que ha comptat tot sumant el 0 que es constant ja en el vector.



7.1.5.7. Diagrama de flux de la funció Long_granja()

Aquesta funció s'encarrega de comptar el nombre de components que presenta el vector Long_granja. Recorre component a component i compta el nombre d'espais en blanc. Finalment, resta els espais de la longitud màxima definida. I el resultat és la longitud del vector.



7.2. Organització dels pins del microcontrolador

En aquest apartat si presenta les diferents connexions entre pins del microcontrolador amb altes components/xips.

El microcontrolador és el cervell del que en l'esquema de la figura 1 està nombrat com a prototip.

Aquest prototip estarà situat de forma pròxima a les sitges, ja que del sensor amb aquest ha d'anar cablejat i és interessant aconseguir reduir la longitud d'aquest cable.

Les següents taules són les diferents connexions i la nomenclatura utilitzada en el codi.

SISTEMA D' ENVIAMENT DE DADES			
SIGFOX		**SIM CARD**	
NOM	**NÚMERO DE PIN**	**NOM**	**NÚMERO DE PIN**
TEST_PIN	6	SIM.card(Rx)	10
SIGFOX_RESET_PIN	7	SIM.card(Tx)	11
SERIAL2(RX)	17		
SERIAL2(TX)	16		

Taula 1: Nomenclatura i pins del microcontrolador segons el sistema d'enviament de dades.

SISTEMA DE MESURA DE DADES			
CÈL·LULA DE CARGA		**SENSOR RADAR/ULTRASÒNIC**	
NOM	**NÚMERO DE PIN**	**NOM**	**NÚMERO DE PIN**
ReDe_Pin	49	Triger	3
SERIAL1 (Tx)	18	Echo	4
SERIAL1 (Rx)	19		

Taula 2: Nomenclatura i pins del microcontrolador segons el sistema de mesura de dades.

7.3. Disseny del circuit i del PCB

Per la realització del PCB, s'ha utilitzat el programa Ares, que és un afegit del ISIS Professional altrament conegut com a Proteus.

Per la realització d'aquest s'han seguit dos processos principals.

- Primerament, s'ha creat el disseny gràfic i entenedor amb la interfície gràfica que ens proporciona el ISIS i s'ha guardat aquest com a component.
- Finalment, s'ha utilitzat el disseny previ per a l'elaboració del PCB amb Ares.

7.3.1. Creació del esquema electrònic del prototip

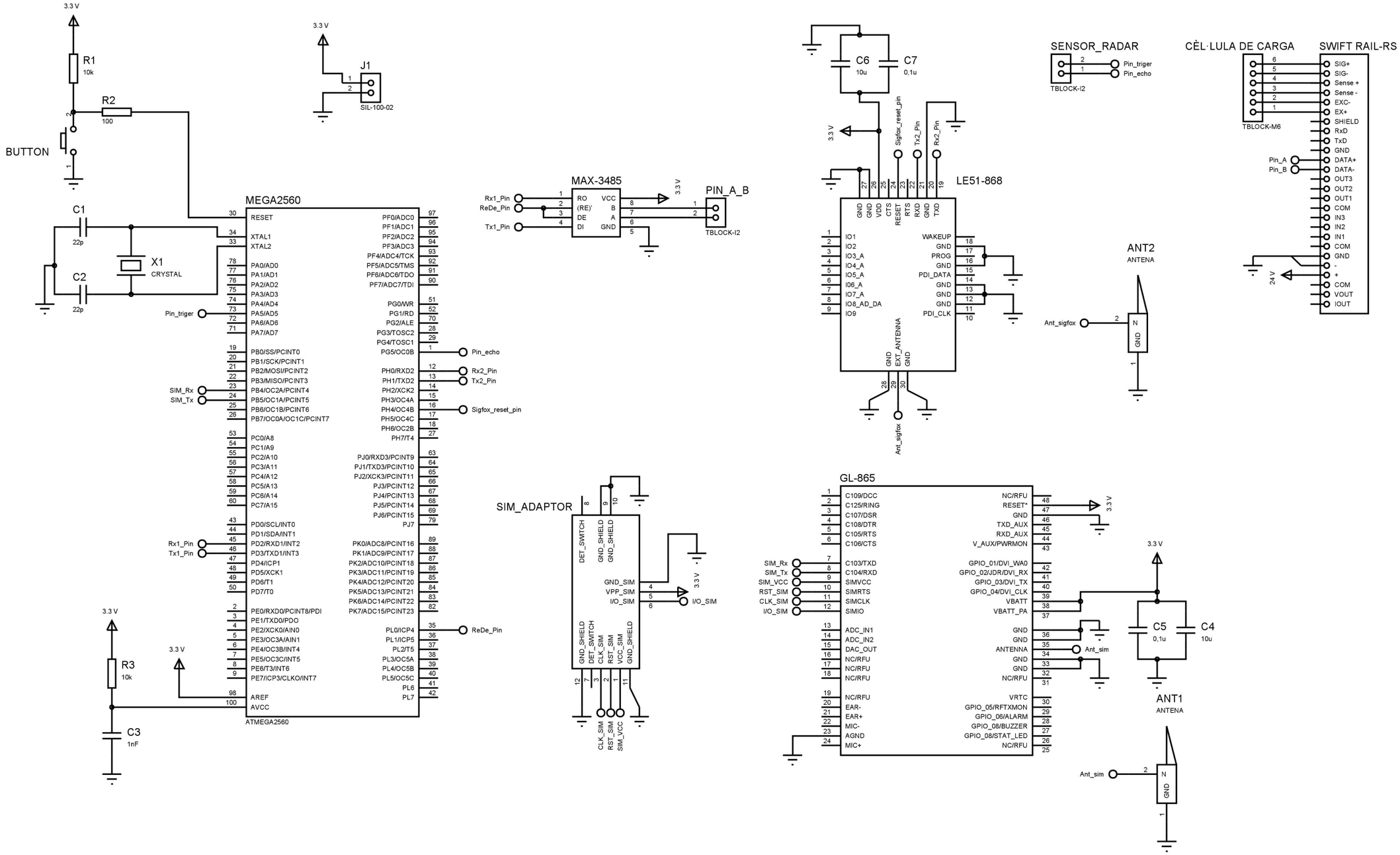
En aquesta secció s'ha elaborat el disseny on es veuen clarament tots i cadascun dels components que són necessaris pel funcionament dels dos prototips en un de sol.

Per aquest projecte s'han tingut de crear les interfícies gràfiques de certs components que no eren existents per defecte, en el propi programa. S'han creat:

- MAX-3485.
- LE51-868.
- Adaptador de la targeta SIM.
- GL-865.
- Swift Rail-RS.
- Antenes.

Un cop realitzats, s'han fet les corresponents connexions amb pins en lloc de cables. Així visualment és més estètic i entenedor. A efectes pràctics, la connexió és idèntica.

En la següent pàgina es pot veure el disseny complet del que s'ha anomenat prototip.

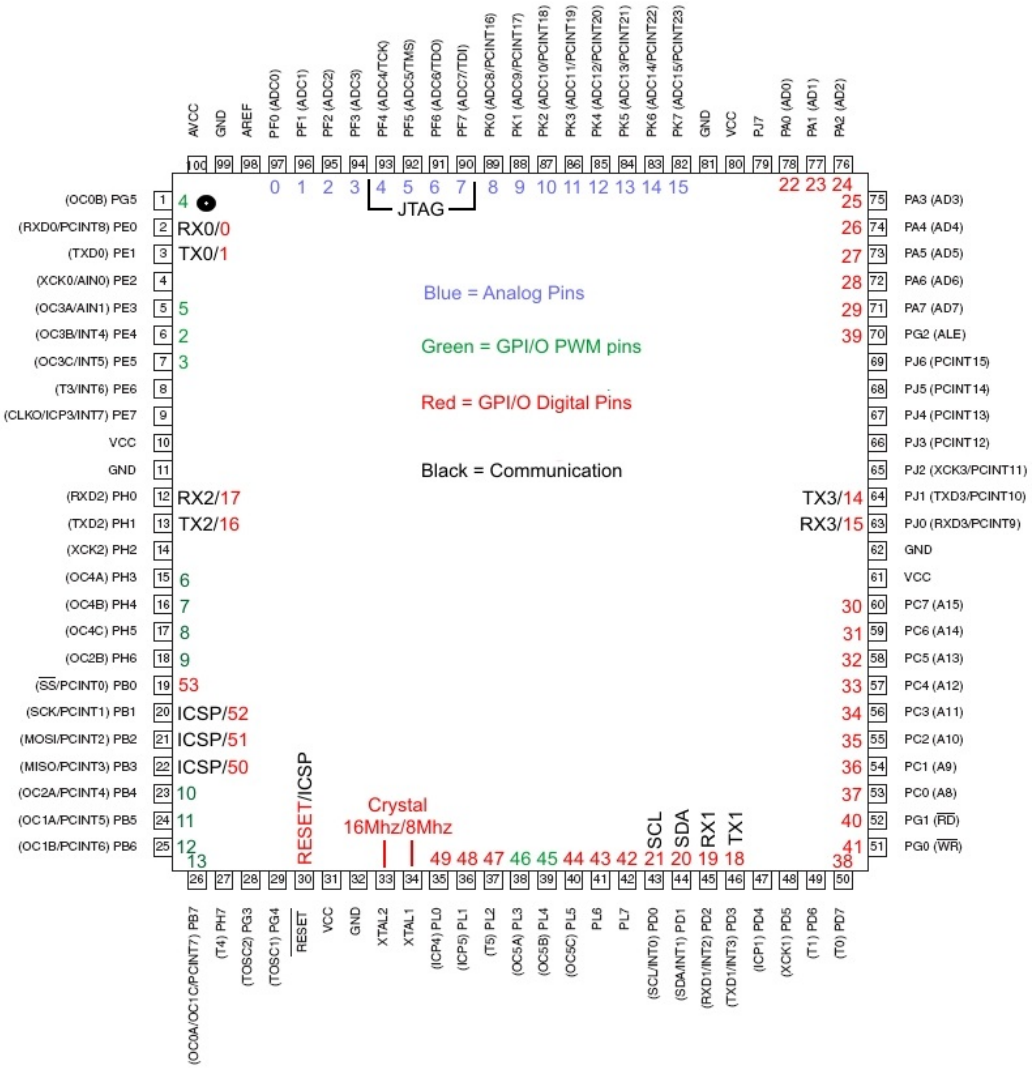

3.3 V
R1
10k
R2
100
BUTTON
J1
SIL-100-02
MEGA2560
C1
22p
C2
22p
X1
CRYSTAL
RESET
XTAL1
XTAL2
PA0/AD0
PA1/AD1
PA2/AD2
PA3/AD3
PA4/AD4
PA5/AD5
PA6/AD6
PA7/AD7
Pin_triger
PB0/SS/PCINT0
PB1/SCK/PCINT1
PB2/MOSI/PCINT2
PB3/MISO/PCINT3
PB4/OC2A/PCINT4
PB5/OC1A/PCINT5
PB6/OC1B/PCINT6
PB7/OC0A/OC1C/PCINT7
SIM_Rx
SIM_Tx
PC0/A8
PC1/A9
PC2/A10
PC3/A11
PC4/A12
PC5/A13
PC6/A14
PC7/A15
PD0/SCL/INT0
PD1/SDA/INT1
PD2/RXD1/INT2
PD3/TXD1/INT3
PD4/ICP1
PD5/XCK1
PD6/T1
PD7/T0
Rx1_Pin
Tx1_Pin
PE0/RXD0/PCINT8/PDI
PE1/TXD0/PDO
PE2/XCK0/AIN0
PE3/OC3A/AIN1
PE4/OC3B/INT4
PE5/OC3C/INT5
PE6/T3/INT6
PE7/ICP3/CLKO/INT7
AREF
AVCC
ATMEGA2560
PF0/ADC0
PF1/ADC1
PF2/ADC2
PF3/ADC3
PF4/ADC4/TCK
PF5/ADC5/TMS
PF6/ADC6/TDO
PF7/ADC7/TDI
PG0/WR
PG1/RD
PG2/ALE
PG3/TOSC2
PG4/TOSC1
PG5/OC0B
Pin_echo
PH0/RXD2
PH1/TXD2
Rx2_Pin
Tx2_Pin
PH2/XCK2
PH3/OC4A
PH4/OC4B
Sigfox_reset_pin
PH5/OC4C
PH6/OC2B
PH7/T4
PJ0/RXD3/PCINT9
PJ1/TXD3/PCINT10
PJ2/XCK3/PCINT11
PJ3/PCINT12
PJ4/PCINT13
PJ5/PCINT14
PJ6/PCINT15
PJ7
PK0/ADC8/PCINT16
PK1/ADC9/PCINT17
PK2/ADC10/PCINT18
PK3/ADC11/PCINT19
PK4/ADC12/PCINT20
PK5/ADC13/PCINT21
PK6/ADC14/PCINT22
PK7/ADC15/PCINT23
PL0/ICP4
ReDe_Pin
PL1/ICP5
PL2/T5
PL3/OC5A
PL4/OC5B
PL5/OC5C
PL6
PL7
MAX-3485
RO
(RE)'
DE
DI
VCC
B
A
GND
PIN_A_B
TBLOCK-I2
C6
10u
C7
0,1u
LE51-868
GND
VDD
CTS
RESET
RTS
RXD
TXD
IO1
IO2
IO3_A
IO4_A
IO5_A
IO6_A
IO7_A
IO8_AD_DA
IO9
WAKEUP
PROG
PDI_DATA
PDI_CLK
EXT_ANTENNA
SENSOR_RADAR
Pin_triger
Pin_echo
TBLOCK-I2
CÈL·LULA DE CARGA
SWIFT RAIL-RS
TBLOCK-M6
SIG+
SIG-
Sense +
Sense -
EXC-
EX+
SHIELD
RxD
TxD
DATA+
DATA-
OUT3
OUT2
OUT1
COM
IN3
IN2
IN1
VOUT
IOUT
Pin_A
Pin_B
24 V
ANT2
ANTENA
Ant_sigfox
SIM_ADAPTOR
DET_SWITCH
GND_SHIELD
GND_SIM
VPP_SIM
I/O_SIM
CLK_SIM
RST_SIM
VCC_SIM
R3
10k
C3
1nF
GL-865
C109/DCC
C125/RING
C107/DSR
C108/DTR
C105/RTS
C106/CTS
C103/TXD
C104/RXD
SIMVCC
SIMRTS
SIMCLK
SIMIO
ADC_IN1
ADC_IN2
DAC_OUT
NC/RFU
EAR-
EAR+
MIC-
AGND
MIC+
RESET*
TXD_AUX
RXD_AUX
V_AUX/PWRMON
GPIO_01/DVI_WA0
GPIO_02/JDR/DVI_RX
GPIO_03/DVI_TX
GPIO_04/DVI_CLK
VBATT
VBATT_PA
ANTENNA
GPIO_05/RFTXMON
GPIO_06/ALARM
GPIO_08/BUZZER
GPIO_08/STAT_LED
C5
0,1u
C4
10u
ANT1
ANTENA
Ant_sim

Pel que fa al component amb nom J1 únicament té la finalitat de permetre una connexió d'un cable a la placa per portar el voltatge de 3,3 V en aquesta.

7.3.1.1. Mega2560

És el microcontrolador de tot el prototip.

S'han connectat únicament les mateixes connexions que s'havien realitzat amb la placa Arduino, i el pins estrictament necessaris.



Il·lustració 42: Equivalència entre pins de l' Arduino Mega 2560 i el microxip Mega 2560.

- El pin numero 30, el reset, és actiu a nivell baix, per tant s'ha posat un polsador que en ser premut posa l'entrada en estat 0 activant així el reset del microcontrolador. S'han utilitzat dues resistències en sèrie (sempre que el polsador no estigui premut) per evitar introduir una corrent massa elevada en el microcontrolador..
- Els pins 33 i 34, corresponen a l'oscil·lador de cristall de 16 MHz, aquest actua com a rellotge de l' Arduino, és a dir, marca una freqüència de funcionament. Per tant, és estrictament necessari.

  Aquest oscil·lador de cristall està posat en paral·lel amb dos condensadors de 22 pF. Aquests dos serveixen per estabilitzar la senyal.
- El pin 98, és el voltatge analògic de referència que el microcontrolador utilitza, en aquest cas és de 3,3 V.
- El pin 100 és el de voltatge. Per tant, a partir d'aquest s'alimenta el microxip. Aquesta entrada està estabilitzada per un condensador i la seva corrent es regulada per la resistència.

  Generalment l' Arduino s'alimenta a 5 V, però aquí solament s'utilitza el microcontrolador, que també podria funcionar amb aquest voltatge, però posteriorment passarà senyals i ordres a altres xips d'alimentació 3,3 V. Per tant, aquesta diferència podria causar problemes, inclús danys en els altres xips. La solució és fer funcionar tots els components al mateix voltatge.
- La resta de pins: 1, 12, 13, 16, 23, 24, 35, 45 i 46 són les connexions que en el procès de disseny estaven connectats a la placa Arduino.

Tots els pins no utilitzats poden ser menyspreats ja què no són necessaris per aquest projecte. Per tant, per a futures modificacions es podrien arribar a agafar altres dades de la granja i ser processades amb aquest mateix microxip realitzant petites modificacions en el disseny.

7.3.1.2. MAX-3485

Aquest dispositiu ja exposat en l'apartat 7.3.5.1

Únicament cal destacar la connexió dels pins 6 i 7, que corresponen a A i B del xip. Aquests són connectats directament a una terminal de bloc, que posteriorment a la placa PCB ens permetrà unir aquests pins a través de cablejat a la corresponent posició del Swift Rail-RS tal i com ja s'ha explicat en apartats anteriors.

7.3.1.3. GL-865

El component encarregat de transmetre les dades a través de la connexió a internet amb la targeta SIM.

Els pins a connectar són:

- El 7 i el 8, són els encarregats de la transmissió/recepció de dades, per tant van connectats al pin corresponent del microcontrolador.
- El 9, 10, 11 i 12 van directament connectats al component que conté la targeta SIM
- El pin 35, va connectat a l'antena.
- Els pins 38 i 39 són l'alimentació del xip. Necessiten 3,3 V. A més a més aquesta senyal és estabilitzada amb dos condensadors, un de 0,1 μF i l'altre amb 10 μF en paral·lel.

  A nivell comercial, aquesta combinació de condensadors és molt comuna i té un funcionament d'estabilització de la senyal força acurada en una gran quantitat de casos pràctics.
- El connector 47 és del reset, que és actiu a nivell baix. És a dir quan hi ha un 0 lògic es reseteja.

  En l'aplicació no m'interessa resetejar el xip perquè des del codi introduït al microcontrolador ja se tanca les connexions amb la xarxa. Per tant, directament s'hi connecten 3,3 V, posant així un 1 lògic.
- Finalment, els pins 35, 36 i 46 van directament a terra.

7.3.1.4. SIM_ADAPTOR

Component que en les proves amb la placa Arduino i els corresponents shields no era utilitzat perquè ja estava integrat en aquests.

S'encarregarà d'establir la comunicació entre el GL-865 i la targeta SIM.

- Els pins 1, 2 ,3 i 6 s'han de connectar directament als corresponents pins del xip GL-865.
- Tant el pin 4, 9, 10, 11 i 12 s'han de connectar a terra.
- El 5 és l'alimentació de la targeta SIM, que serà de 3,3 V pel mateix motiu que ja s'ha esmentat en el Mega 2560.

7.3.1.5. LE51-868

Component que es comunica amb la plataforma Sigfox. Les seves connexions són les següents:

- Els pins 19, 21 i 23 van connectats directament al microcontrolador Mega 2560.
- El 29 va directament unit amb la corresponent connexió de l'antena.
- El 25 és l'alimentació. També funciona a 3,3 V, estabilitzats amb la mateixa estructura de condensadors del GL-865.
- El 11, 12, 13, 15, 17, 20, 26 i 27 van directament al terra.

7.3.1.6. Antenes

Aquestes són les encarregades de facilitat la comunicació entre els dispositius encarregats de la comunicació de dades i la xarxa.

Tenen una connexió a terra, el pin 1. Mentre que l'altre al corresponent mòdul, com ja s'ha esmentat.

7.3.1.7. Swift Rail-RS

Els 4 o 6 cables característics de les cèl·lules de carga van connectats directament a les corresponents entrades del Swift Rail-RS. En el propi datasheet del sensor s'indica que és cada cable.

Pel que fa, al Data+ i Data-, com ha estat mencionat, es connectarà de la sortida del PCB amb forma de cable.

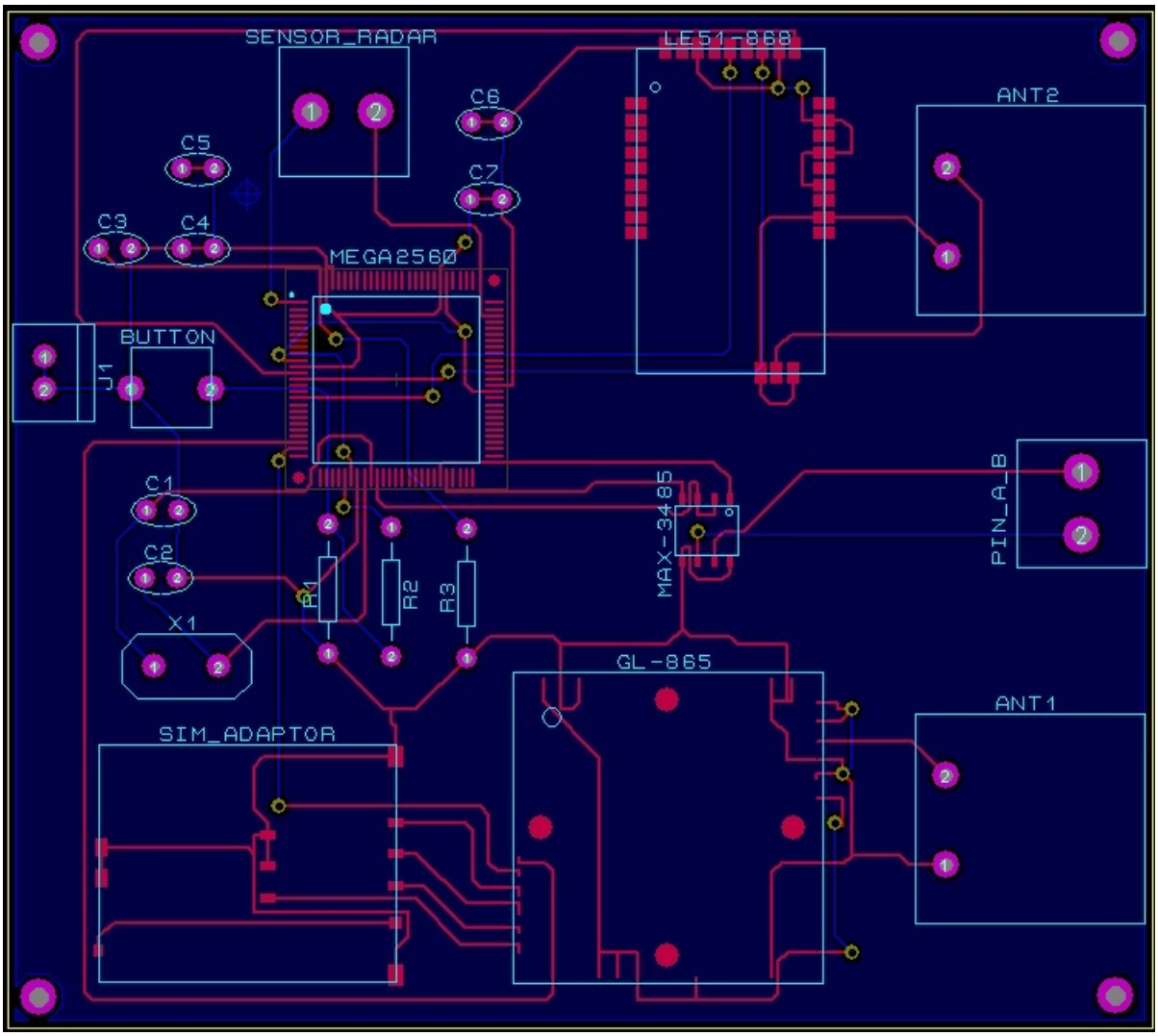
L'alimentació d'aquest és a 24 V.

7.3.2. Disseny del PCB

Per aquest apartat, també s'han tingut de crear les footprints del components no existents en el programa.

Aquestes s'han dimensionat d'acord amb les mides dels datasheets. Això implica que la visualització 3D del PCB final, que el propi programa realitza no és acurat del tot i presenta certs defectes estètics que no tenen cap repercussió en la seva funcionalitat.

Les dimensions de la placa final són d'aproximadament uns 9 x 9 mm. Essent així una mida reduïda i fàcil d'adaptar en qualsevol caixa protectora.



Il·lustració 43: Disseny del PCB del prototip final. En blau els traçats de la capa superior i en vermell els de la capa inferior. Realitzada amb Ares.

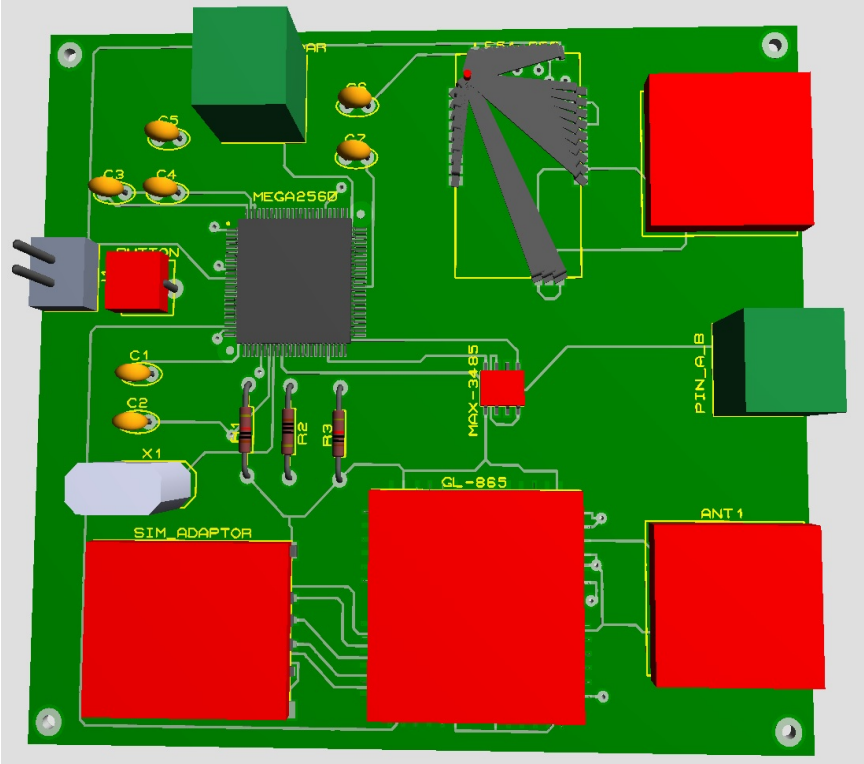
La placa presenta dos nivells. De color blau podem observar totes les traces que es troben impreses en la part superior de la placa i amb vermell les realitzades en la part inferior.

Tots els punts roses són els forats en els quals les potes dels components s'introduiran. I els cercles daurats són els ponts, és a dir, comunicacions entre la part inferior i superior de la placa.

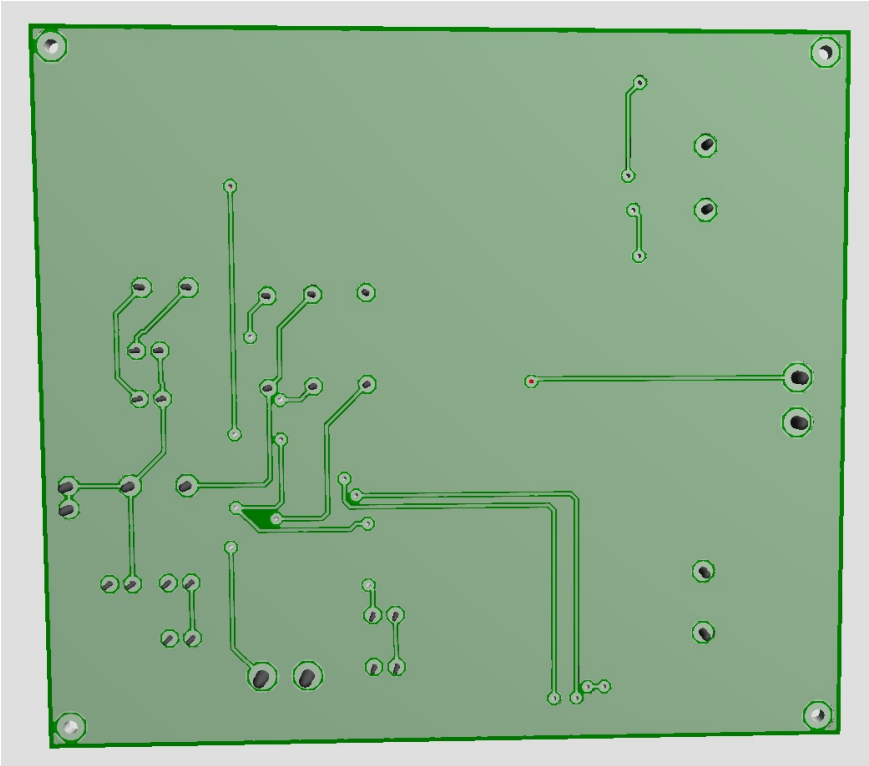
Pel que fa a la part inferior, s'ha posat l'opció de GND=POWER. Uneix totes les línies que són terra amb la pròpia placa. Això serveix per evitar gran quantitat de sorolls i senyals que distorsionin la senyal.

L' impressió d'aquesta placa té un pressupost aproximat d'uns 75-90 €.

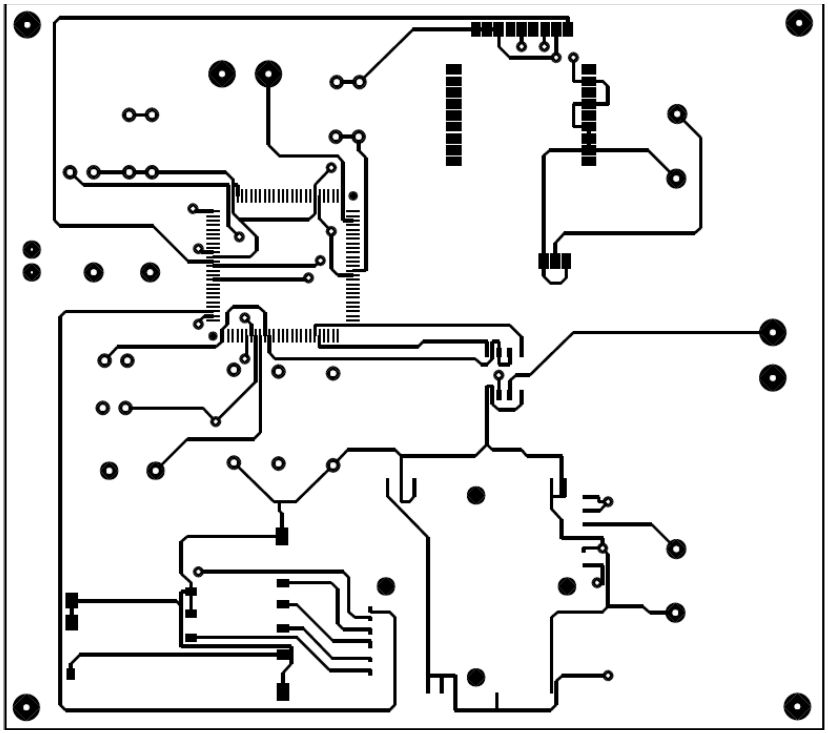
Les següents figures (44-45) són una visualització amb 3D que el propi programa realitza, per tenir una idea aproximada de com quedara la placa final, tant per la part superior con inferior. També hi ha les plantilles d'impressió de la part frontal i la del darrera, que són les figures (46-47). I la figura 48 és informació addicional a imprimir sobre la capa davantera de la PCB.



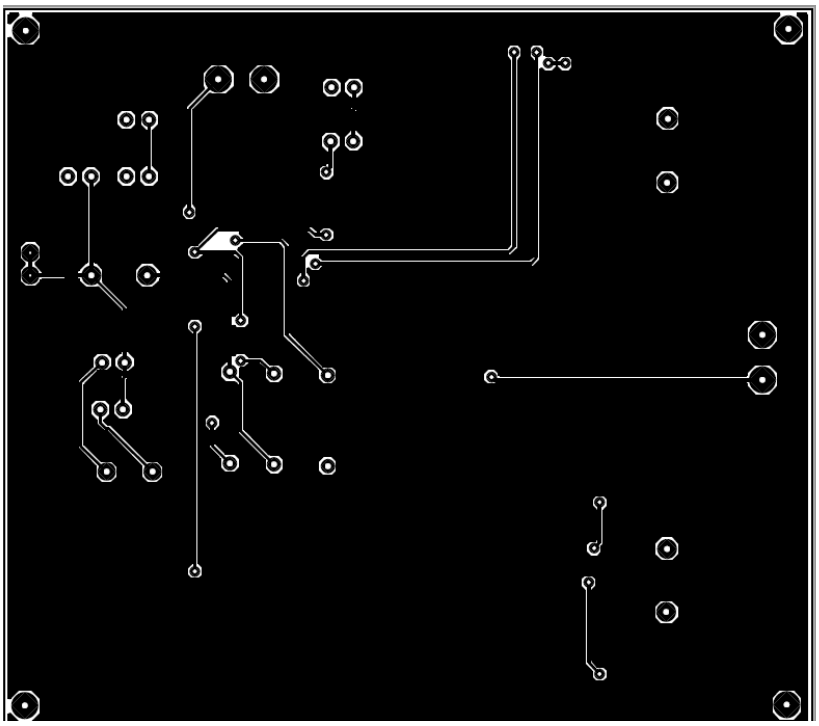
Il·lustració 44: Visualització 3D de la placa PCB. Part davantera. Realitzada amb Ares.



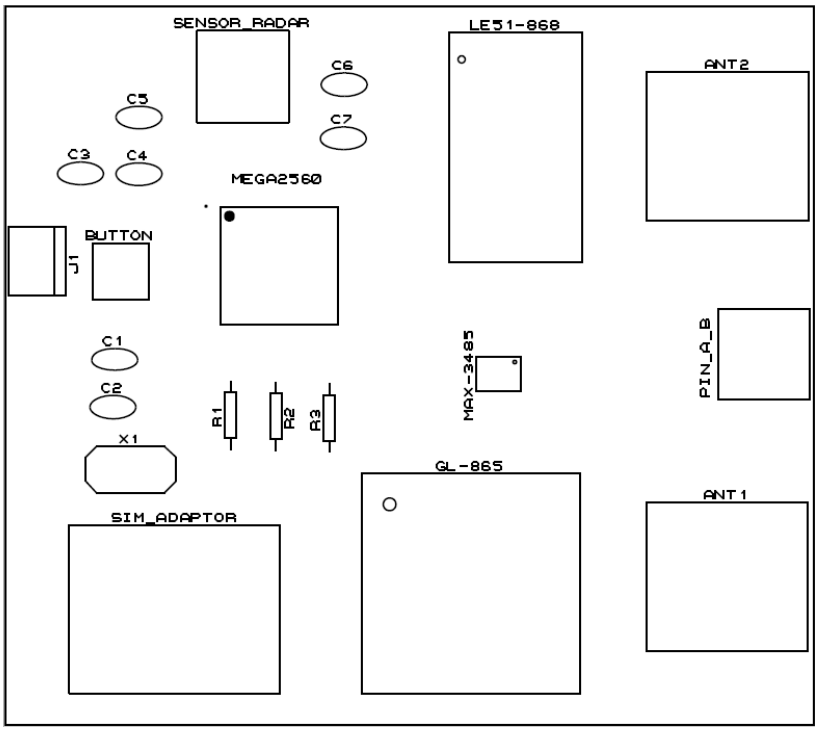
Il·lustració 45: Visualització 3D de la placa PCB. Part trasera. Realitzada amb Ares.



Il·lustració 46: Impressió frontal del PCB.



Il·lustració 47: Impressió trasera del PCB.



Il·lustració 48: Informació a adjuntar a l' impressió davantera del PCB.

8. Calibratge dels sensors de mesura

El prototip necessita rebre informació per enviar. Aquesta informació és captada pels sensors que van directament col·locats a la sitja. Aquests sensors necessiten un calibratge per tal de donar una mesura correcta al microcontrolador.

8.1. Sensor radar/ultrasònic

Pel que fa al sensor d'ultrasons, tal i com en el principi físic s'ha explicat detalladament, podem trobar la distància entre el sensor i el producte. Doncs bé, un cop tenim aquesta distància, si es pot obtenir una expressió matemàtica que ens quantifiqui el volum restant en el recipient de contenció, en funció de la distància, es pot obtenir la massa restant.

La massa és proporcional al volum i la densitat del producte.

$$m = \rho \cdot V_{total}$$

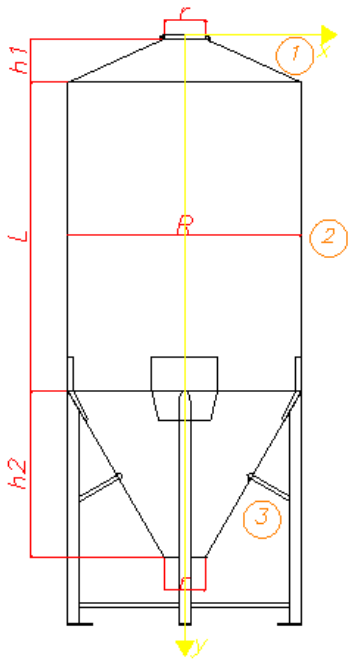
on:

m = massa

ρ = densitat pinso

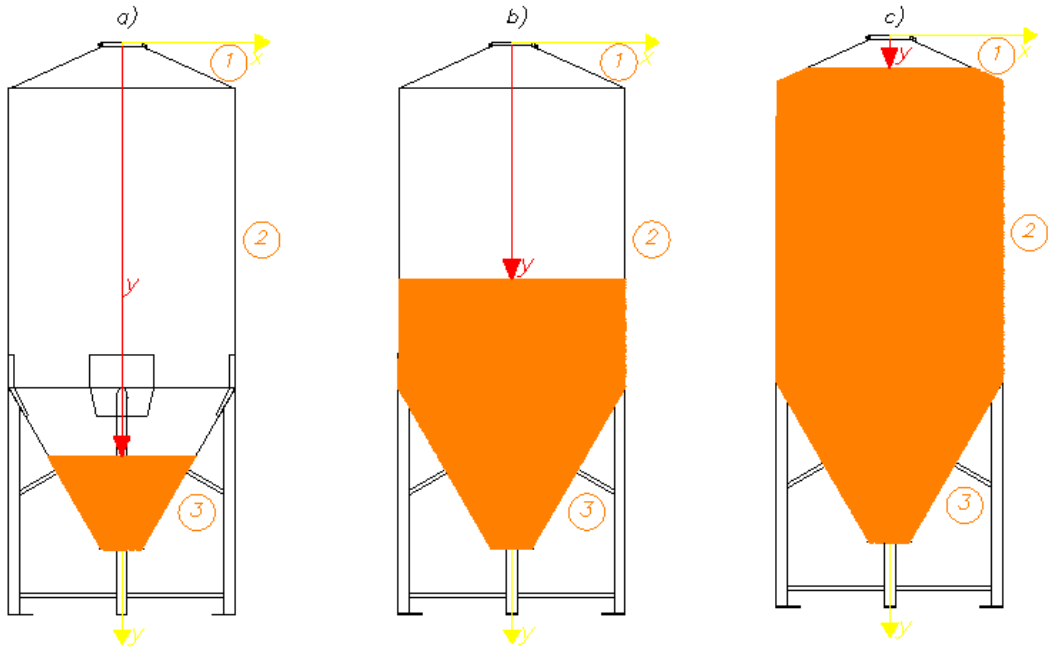
V_{total} = Volum total

Un cop això es necessita tenir present un sistema de nomenclatura per deduir l'expressió matemàtica.



Il·lustració 49: Nomenclatura genèrica d'una sitja de granja.

Hi ha 3 possibles parts a estudiar tenint en compte les 3 formes geomètriques diferents en la sitja i el nivell de pinso que conté.



Il·lustració 50: Casos possibles de nivell amb diferent equació matemàtica.

Quan el pinso esta en el cas 1 , corresponent a la imatge c) la equació resultant de la massa de pinso restant és:

$$m(y)=\rho\cdot[\frac{\pi\cdot h_2}{12}\cdot(D^2+D\cdot d+d^2)+\frac{\pi\cdot D^2}{4}\cdot L+\frac{\pi\cdot(h_1-y)}{12}\cdot(D^2+D\cdot d+d^2)]$$

Pel cas 2, equivalent a la imatge b) la equació és:

$$m(y)=\rho\cdot[\frac{\pi\cdot h_2}{12}\cdot(D^2+D\cdot d+d^2)+\frac{\pi\cdot D^2}{4}\cdot(L+h_1-y)]$$

I finalment si el pinso es troba en el la regió 3, imatge a), l'equació és:

$$m(y)=\rho\cdot[\frac{\pi\cdot(L+h_1+h_2-y)}{12}\cdot(D^2+D\cdot d+d^2)]$$

On:

ρ=*densitat pinso*

$D=R\cdot 2$

$d=r\cdot 2$

Ara únicament cal introduir aquestes equacions i els paràmetres corresponents al codi del microxip, tot classificant en cada mesura, la part de la sitja en la que es troba el nivell actual de pinso.

8.2. Cèl·lula de carga

Aquestes cèl·lules de carga donen uns valors imprevisibles si es connecten directament al microxip, doncs aquesta informació s'ha de processar. Per tant, s'ha de fer un calibratge per a cada cèl·lula de carga en concret.

Aquest calibratge és fa directament amb el mòdul Swift Rail-RS, el qual ja presenta una sèrie d'instruccions a seguir en el seu manual.

Permet l'opció de calibrar a partir d'una massa coneguda o a partir d'una referència de mV/V.

Per tant, aquesta calibratge s'ha de realitzar en cada cas particular.

Un cop s'ha fet aquesta, ja podem observar pel display del mòdul, el valor de massa restant de forma precisa.

9. Pressupostos

Aquí podem observar els preus de tots els components de forma independent i per unitat.

Aquests preus poden ser molt inferiors degut que l'empresa té previst treballar a grans quantitats, per tant únicament és una estimació del pressupost que s'obtindrà.

PRODUCTE	FABRICANT	PREU(€/UNITAT)
DR-15-24	Mean Well	16,29
RS-15-3,3	Mean Well	8,32
MAX 3485	Maxim Integrated	2,80
LE51-868-S	Telit	10 (aprox)
GL865-QUAD	Telit	12 (aprox)
ATMEGA2560	Atmel	9,64
SWIFT RAIL RS	Uticell	248
CAIXA SUMA	Uticell	37
PLACA PCB	2 CI	75
SV3000	Dinacell	138,60
Vega Puls69	Vega	645
Condensador de 10 µF	RS Components	0,83
Condensador de 22 pF	RS Components	0,70
Condensador de 0,1 µF	RS Components	0,34
Condensador de 1 nF	RS Components	0,19
Resistència de 0,1 kΩ	RS Components	0,039
Resistència de 10 kΩ	RS Components	0,045

Taula 3: Components amb la seva marca de fabricant i el preu unitari.

I pel que fa al tipus de servei usat de transmissió:

TIPUS DE TRANSMISSIÓ	PREU (€)
Sigfox	1 €/any
Targeta SIM	0,7 €/mes

Taula 4: Tipus de transmissió i preu.

Per tant, podem distingir 4 tipus de pressupostos diferents en funció de les característiques del prototip.

Però d'aquest únicament se distingiran en funció del cost del producte, sense tenir en compte la plataforma, ja que és un cost de tipus constant, mentre que el valor del producte solament és pagarà una vegada.

9.1. Pressupost prototip amb cèl·lula de carga.

COMPONENT	UNITATS	PREU(€)
DR-15-24	1	16,29
RS-15-3,3	1	8,32
MAX 3485	1	2,80
ATMEGA2560	1	9,64
SWIFT RAIL RS	1	248
CAIXA SUMA	1	37
PLACA PCB	1	75
SV3000	4	138,60
Condensador de 10 µF	2	1,66
Condensador de 22 pF	2	1,40
Condensador de 0,1 µF	2	0,68
Condensador de 0,1 nF	1	0,19
Resistència de 0,1 kΩ	1	0,039
Resistència de 10 kΩ	2	0,045
	COST TOTAL	955,46

Taula 5: Pressupost del prototip amb cèl·lula de carga.

Per tant aquí si ha d'afegir el cost del dispositiu de transmissió Sigfox o bé targeta SIM.

El cost final actual del prototip amb cèl·lula de carga és de:

NOU-CENTS CINQUANTA-CINC EUROS AMB 46 CÈNTIMS.

9.2. Pressupost prototip amb sensor radar.

COMPONENT	UNITATS	PREU(€)
DR-15-24	1	16,29
RS-15-3,3	1	8,32
MAX 3485	1	2,80
ATMEGA2560	1	9,64
PLACA PCB	1	75
VEGA PULS 69	1	645
Condensador de 10 µF	2	1,66
Condensador de 22 pF	2	1,40
Condensador de 0,1 µF	2	0,68
Condensador de 0,1 nF	1	0,19
Resistència de 0,1 kΩ	1	0,039
Resistència de 10 kΩ	2	0,045
	COST TOTAL	761,064

Taula 6: Pressupost prototip amb sensor radar.

Per tant aquí si ha d'afegir el cost del dispositiu de transmissió Sigfox o bé targeta SIM.

El cost final actual del prototip amb sensor radar és de:

SET-CENTS SEIXANTA-UN EUROS AMB 7 CÈNTIMS.

9.3. Valoració dels prototips amb funció dels costos

Tal i com es pot observar en les taules anteriors hi ha una diferència relativament considerable, d'uns 194 € entre prototips. El disseny amb cèl·lules de carrega és el més car.

Aquests valors econòmics són considerant la compra de components a quantitats mínimes i exactes per 1 prototip. Aquest fet fa augmentar el preu de forma exponencial. L'empresa té una necessitat important, i pretén aplicar el disseny a un gran nombre de sitges. Això és altament valorat per les empreses proveïdores i el descompte que rep la companyia és molt important i considerable.

Paral·lelament, s'ha de valorar que el prototip amb sensor radar necessita una adaptació del codi en cada cas. Això és pot traduir a nivell econòmic com temps dels empleats per la modificació de codi i la possibilitat d'error humà, que influència directament al projecte d'anàlisis.

Per tant, valorant tots els factors, considero que el preu real final dels dos prototips és equivalent i que és favorable escollir el de les cèl·lules de carga.

BLOC III - IMPLEMENTACIÓ REAL

10. Prototip inicial creat

Després de realitzar tot l'estudi i proves pertinents s'ha arribat amb aquest prototip totalment funcional.

Actualment, es troba posat en una caixa de protecció per simular el recipient de contenció final.

Totes els components estan units mitjançant soldadures d'estany i pins amb puntes específiques per aquest tipus de connexió.

10.1. Exemple de funcionament

Inicialment, és modifica els paràmetres característics del codi, que en són:

- Tipus de sensor.
- Nom de la granja.

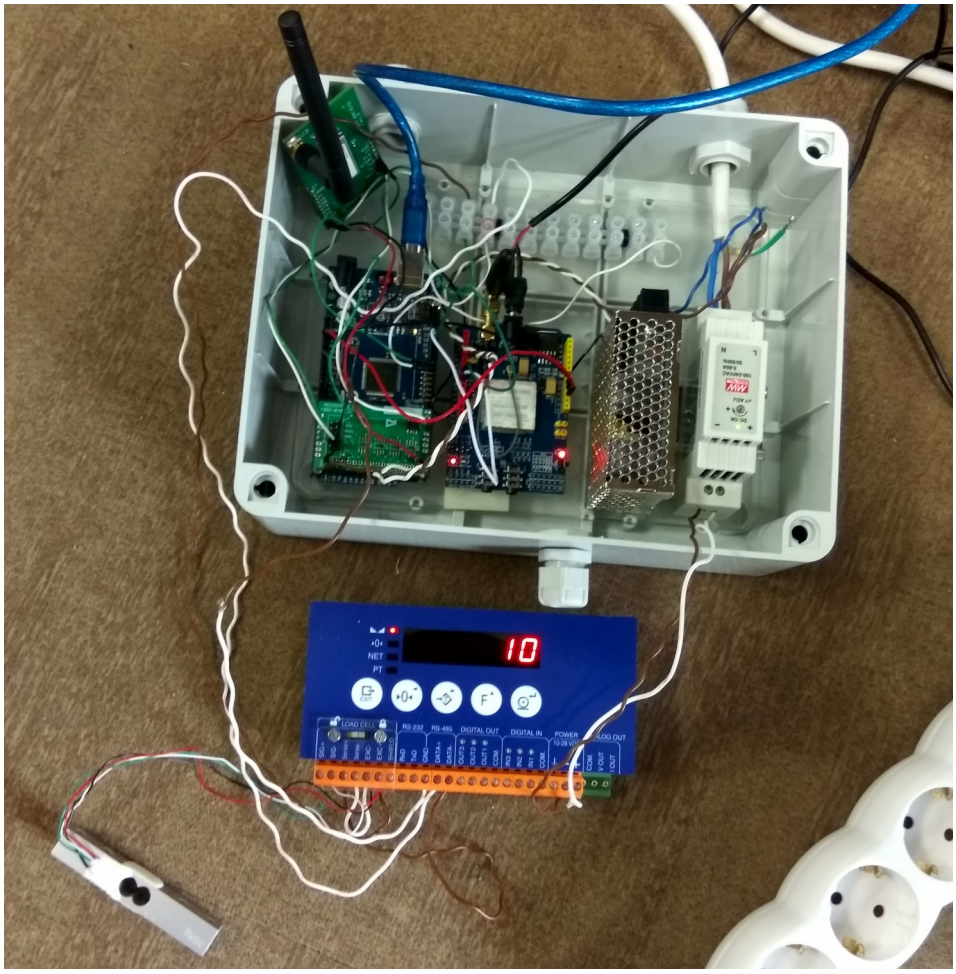
En aquest cas s'ha usat una cèl·lula de carga de petita capacitat, concretament de 20 Kg de tracció. Però això no és significatiu perquè el principi de funcionament d'una cèl·lula de major capacitat és el mateix, mentre que el preu és molt superior.

I pel nom de la granja, s'ha posat un qualsevol com pot ser «25-Artesa».

```
//PROGRAMA A INTRODUIR EN EL MICROXIP, PROTOTIP "SIM CARD".
//_____________________________________DADES A CANVIAR(solament modificar el que està dins d'aquest apartat_______

#define SENSOR_LECTURA "cel_carga"   // ¡¡¡ CANVIAR EN FUNCIÓ DEL SENSOR DE MESURA, "cel_carga" O BÉ "radar". !!!
char granja[25]="25-Artesa";  // ¡¡¡ CANVIAR PEL NOM DE LA GRANJA AMB QÜESTIÓ, NO HI POT HAVER CAP ESPAI EN BLANC. !!!
```

Il·lustració 51: Dades a modificar del codi en l'exemple mostrat.

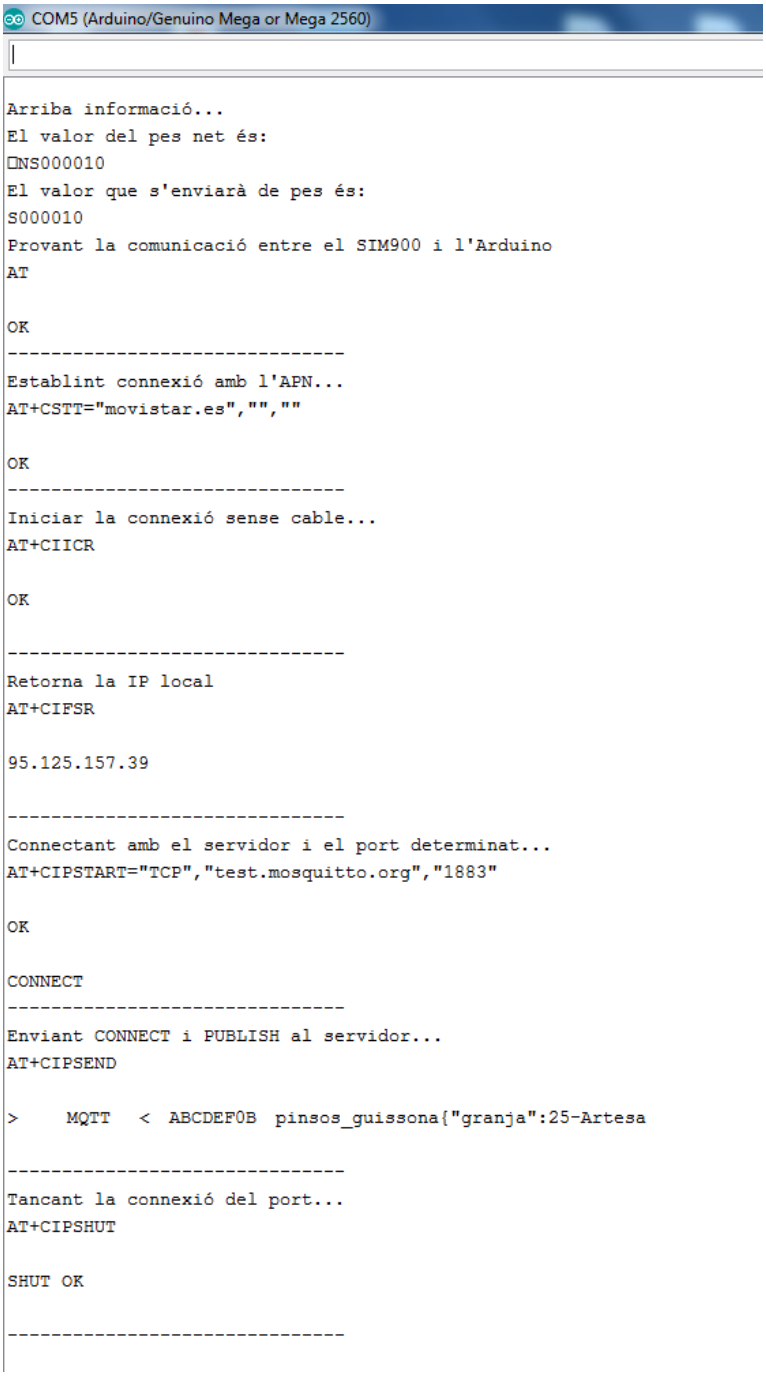


Il·lustració 52: Exemple d'utilització del prototip final.

Es pot observar que el Swift Rail-RS marca 10 Kg. Estrictament parlant, no és una mesura real, perquè la cèl·lula de carga no presenta cap pes. Això és degut a que no està calibrat el mòdul de bàscula. Però a efectes pràctics i de visualització del funcionament és molt interessant tenir un valor diferent de 0, ja què en realitat difícilment arribarà amb aquest. Motiu pel qual he deixat aquest valor.

Per tant, d'acord amb la informació presentada en apartats anteriors, la sortida que s'espera és: S000010 perquè el pes és una mesura estable i el valor que mostra es 10.

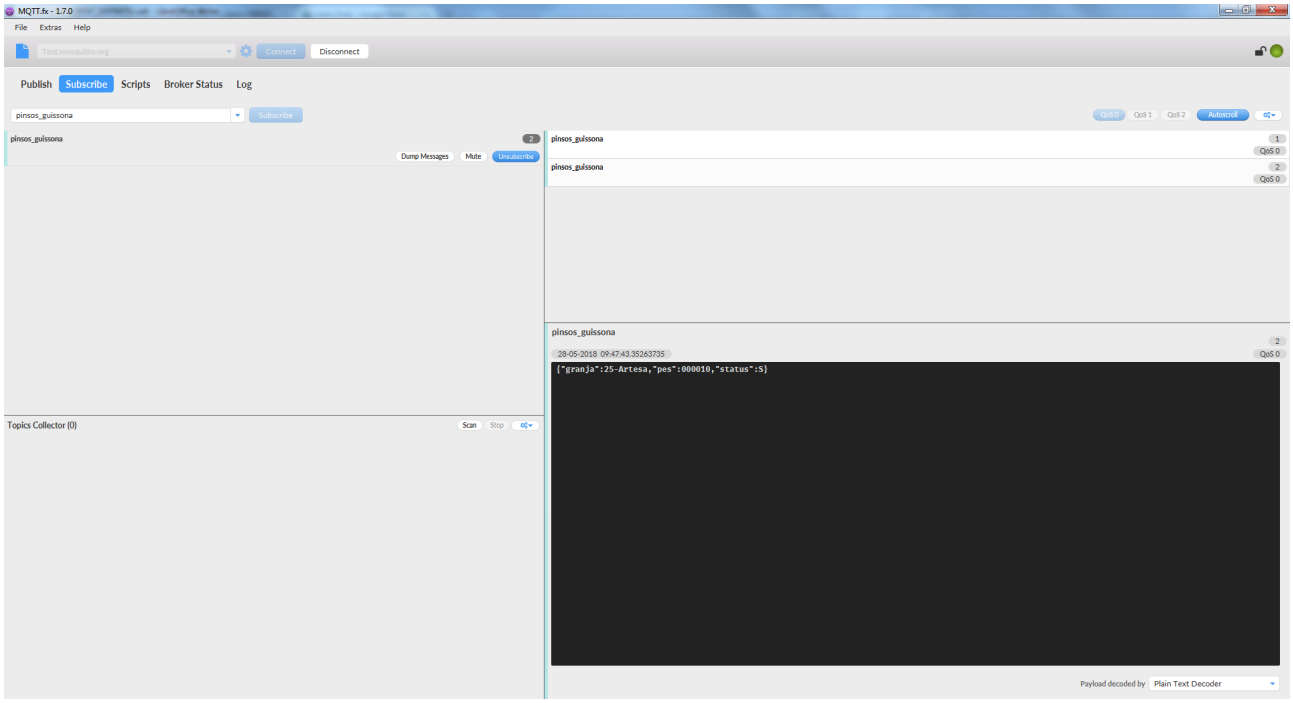
Seguidament, s'obra la pantalla de sortida del microcontrolador per poder seguir el procès que aquest realitza a temps real. El resultat final és el següent:



Il·lustració 53: Pantalla de sortida del procès del microcontrolador.

Es pot observar que les respostes que és reben del les comandes AT són les esperades, tal i com s'ha esmentat en un apartat anterior.

Finalment, cal comprovar que la dada ha arribat al servidor.



Il·lustració 54: Captura de pantalla del servidor al qual s'ha publicat la dada mesurada.

Aquí s'observa que s'ha enviat correctament i ha quedat emmagatzemada amb el format Json que en el seu corresponent apartat es va introduir.

Ara aquestes dades s'anirien emmagatzemant de forma continuada i acumulativa. En l'exemple hi ha emmagatzemades ja 2 dades...i així successivament es van acumulant de forma indefinida.

És pot dir que aquestes dades ja estan preparades per a ser processades i utilitzades amb l'objectiu que inicialment, l'empresa ja tenia.

11. Conclusions

En aquest treball s'ha realitzat un projecte real d'empresa. Aquesta companyia té la necessitat d'obtenir dades de les sitges de les granges. Com a tal, portar un control del pinso restant permetrà fer unes previsions en la producció d'aquest i així maximitzar els beneficis.

El principal problema a solucionar ha estat la desfavorable geolocalització de les granges, que provoca gran quantitat d'errors en l'enviament de dades, i fa pràcticament impossible utilitzar un únic sistema. Per tant, en aquest treball s'ha estudiat i dissenyat un prototip capaç de funcionar amb dos sistemes d'enviament de dades diferents.

En el moment d'iniciar el treball l'empresa ja havia realitzat algun que altre pas per l'enviament de dades amb Sigfox, que seria una comunicació LPWAN, però el resultat obtingut no era l'esperat.

Doncs, la recerca va iniciar-se buscant les altres opcions existents per enviar dades de forma telemàtica i les úniques opcions factibles eren o bé a través de l' internet d'una targeta SIM o mitjançant el LoRa. Finalment vaig descartar l'última opció perquè necessitava d'una xarxa d'antenes per tot el perímetre, altrament dit, presentava un cost elevadíssim. Des d'aquest moment hem vaig focalitzar totalment en estudiar l'opció m2m amb targeta SIM.

Seguidament vaig buscar els components necessaris per fer el prototip. Escollint primer el microcontrolador i el xip de transmissió de dades. Seguidament, la resta de components necessaris d'acord amb les característiques d'aquests.

Un cop els components clars, va iniciar-se la programació del codi del microcontrolador. Va ser necessària una programació abaix nivell. S'han utilitzat comandes AT, com la telefonia mòbil solia utilitzar als seus inicis juntament amb el protocol MQTT per l'enviament d'informació a una base de dades.

El principal problema que m'ha sorgit ha estat solucionat amb el protocol MQTT que ens permet establir un bypass, esquivant els tallafocs protectors de la xarxa de l'empresa. Inicialment, vaig intentar saltar-los però amb el poc potencial del microcontrolador però va resultar impossible.

Finalment, tot el prototip al conjunt funciona segons el previst.

Pel que fa al preu final del prototip, escollint qualsevol de les dues opcions, ja sigui amb cèl·lula de carga o sensor radar, independentment de la plataforma de transmissió, és més car que el comercial de mesura esmentat en els objectius. Però amb una producció a gran escala, els preus serien aproximadament similars i tindria l'avantatge que aquest prototip permet a l'empresa emmagatzemar les dades.

Pel que fa a la possibilitat de ser aplicat considero que és totalment factible. El prototip compleix les condicions inicials i ha solucionat la problemàtica presentada. També mencionar que el cost final si

es produeix a gran escala serà molt inferior. El principal cost del prototip són els sensors de mesura, que comprats a gran escala, el preu es redueix de forma molt considerable.

Personalment, veig una gran possibilitat d'ús, ja no solament amb l'àmbit del pinso, sinó amb tot el sector d'enviament de dades de forma telemàtica sense una xarxa local. Per tant, crec que aquest treball pot ser una introducció al món de l'emmagatzematge i processament de dades.

12. Propostes de futur

En aquest projecte, les dades són agafades d'una única sitja concreta. El Swift Rail RS permet controlar diverses mesures al mateix temps degut a la diversitat de canals que presenta. Per tant, se podria arribar a obtenir dades de diverses sitges amb un únic Swift Rail RS, fent el prototip funcional per mes d'una sitja.

També crec que amb la gran quantitat de pins disponibles del microcontrolador Mega-2560 és poden controlar molts més aspectes. Entre ells, s'ha esmentat un control de les vegades que els porcs utilitzen els abeuradors. Això permet un control indirecte de la salut dels animals.

De la mateixa forma, es podria arribar a comptabilitzar les vegades que mengen i així tenir un altre tipus de previsió del benestar de la granja.

Aquest projecte, únicament acaba de començar. L'empresa s' introdueix poc a poc al encara desconegut món del Big data i com a tal, hi ha molt a investigar i millorar. Espero que aquesta tesis sigui un inici per encaminar el llarg procés.

13. Bibliografia i referències

[0] https://www.wivity.com/blog/choosing_connectivity_part₂

[1] https://www.sigfox.com/en

[2] https://www.semtech.com/technology/lora

[3] https://www.lora-alliance.org/what-is-lora

[4] https://www.emnify.com/2016/12/01/machine-to-machine-sim-explained/

[5] Instrumentación-Electronica-Thomson-Parte-1 de Miquel A. Pérez Garcia, Juan C. Álvarez Antón, Juan C. Campo Rodríguez, Fco. Javier Ferrero Martín i Gustavo J.Grillo Ortega

[6] http://www.dinacell.com/fichatecnica/SV-3000%20(2986-05)%20Ft.pdf

[7] https://www.utilcell.es/wp-content/uploads/2016/07/Es-En_DS_CAJAS-SUMA_Utilcell.pdf

[8]https://www.vega.com/DocumentDownloadHandler.ashxdocumentContainerId=6554&languageId=6&fileExtension=pdf&softwareVersion=&documentGroupId=47249

[9] http://www.artbrno.cz/download/telit/Telit_LE51-868_S_RF_Module_User_Guide_r1.pdf

[10]https://www.telit.com/wpcontent/uploads/2017/09/Telit_GL865DUAL_QUAD_V3_Product_Description_r6.pdf

[11] http://simcom.ee/modules/gsm-gprs/sim900/

[12] https://www.microchip.com/wwwproducts/en/ATmega2560

[13] https://www.utilcell.es/wp-content/uploads/2016/07/Es-En_DS_SWIFT_Utilcell.pdf

[14] https://www.mouser.es/datasheet/2/260/DR-15-spec-1108947.pdf

[15] http://meanwellusa.com/productPdf.aspx?i=1#1

[16] http://www.alldatasheet.com/datasheet-pdf/pdf/73221/MAXIM/MAX3485.html

[17]https://www.telit.com/wpcontent/uploads/2017/09/Telit_AT_Commands_Reference_Guide_r24_B.pdf

[18]http://mqtt.org/

14. Annex

En aquest apartat hi ha present el codi introduït al microcontrolador i el càlcul de volum del silo emprat anteriorment per la calibració del sensor Vega Puls 69

14.1. Codis del microxip MEGA 2560

```
//_____________________________________________DADES A CANVIAR(solament modificar el que està dins d'aquest apartat_____________________________________________

#define SENSOR_LECTURA "cel_carga"  // ¡¡¡ CANVIAR EN FUNCIÓ DEL SENSOR DE MESURA, "cel_carga" O BÉ "radar". !!!
char granja[25]="25-Artesa";  // ¡¡¡ CANVIAR PEL NOM DE LA GRANJA AMB QÜESTIÓ, NO HI POT HAVER CAP ESPAI EN BLANC. !!!
//_____________________________________________________________________________________________________________

#include <SoftwareSerial.h>

SoftwareSerial SIM_card(10,11); //Rx Tx //mega 10 i 11  Definiex l'objecte SIM_card(Rx,Tx) que és de tipus SoftwareSerial.
int longitud; //variable global que emmagatzema la longitud del vector pub[].
int longitud_granja;  //Variable global que emmagatzema la longitud del vector que emmagatzema el nom de la granja.

int ReDe_Pin=49; //Pin que controlarà l'estat dels pins (RE)' i DE.(HIGH= Emisor / LOW=Receptor)

int Pin_triger=3; //defineix el pin digital número 3 com a triger. El triger és on s'envia la senyal d'ultraso.
int Pin_echo=4; //defineix el pin digital número 4 com a echo. El echo és on es reb la senyal d'ultraso.

char pes_enviar[7]; //El pes que s'enviarà a través del mòdul de SIGFOX o M2M SIM Card.
char pub[50]={0x30,27,0x00,0x0F,0x70,0x69,0x6E,0x73,0x6F,0x73,0x5F,0x67,0x75,0x69,0x73,0x73,0x6F,0x6E,0x61,0x7B,0x22,0x67,0x72,0x61,0x6E,0x6A,0x61,0x22,0x3A};
//És el vector inicial de "publish", posteriorment es modifica en el codi.

void setup() //Configuració del sistema.
{
  Serial.begin(9600); //Obra el serial, és la pantalla de l'ordinador, i configura la velocitat d'enviament de dades a 9600 bps.
```

```
  Serial1.begin(9600);  //Obra el serial1, és a través del qual s'estableix la comunicació RS-485. Pins(18,19)(Tx,Rx)
  SIM_card.begin(9600);  //inicialitza la velocitat de l'objecte SIM_card (bps) definit previament.

  pinMode(ReDe_Pin,OUTPUT); //Es defineix el pin ReDE_Pin com una sortida.
  pinMode(Pin_triger,OUTPUT); //declara el pin_triger com una sortida del sistema.
  pinMode(Pin_echo,INPUT); //declara el pin_echo com una entrada del sistema.
}


void loop()
{
  if (SENSOR_LECTURA == "cel_carga")  //Dins d'aquest condicional anirà tot el codi que
  {
    Lectura_dades_SWIFT();
  }
  else if (SENSOR_LECTURA == "radar")
  {
    Lectura_dades_Radar();

  }
  else
  {
    Serial.println("ERROR DE TIPUS DE SENSOR, REVISAR...!!!");
    delay(1000);
  }
  delay(3000);
  Enviar_dades();
  delay(60000);
}

void Lectura_dades_SWIFT()
{
  char rebut[30]; //El SWIFT RAIL RS envia les dades en format char, per tant posant el vector de forma char, ja és fa la conversió directament.
  char pes[9];  //El mateix motiu que el vector "rebut"
  int posicio; //Variable que guardarà el valor que correspongui a la posició que més endavant es necessitarà.

  posicio=0;  //A cada volta del bucle s'inicia la variable posició a 0.
  digitalWrite(ReDe_Pin,HIGH);  //L'Arduino actua com a "mestre" i el SWIFT RAIL RS com a "esclau". Per tant s'enviaràn dades del microcontrolador a la bàscula.
  delay(2);   //S'espera 2 microsegons perquè sigui temps suficient a que el Pin 49 passia en estat HIGH.
 //El que és fa en les següents comandes, tal i com en el manual d'ús del SWIFT RAIL ens indica és: "solicitud del peso neto y bruto y el pico actual"
 //Maestro: <Addr> "N" EOT. //Comandes que se l'hi han d'introduir en ASCI
 //SWIFT RAIL RS: <Addr> "N" <status> <net> <gross> <peak> ETX <chksum> EOT / És els valors que et retorna en ASCI també.
```

```
  Serial1.write(0x81);  //S'envia a través del bus de comunicació 485 la comanda pertanyent a l'adreça. És la direcció de comunicació serie dels intrument de mesura.
  //Aquesta és el caràcter ASCII que s'obté sumant 80h al nombre de la direcció. (<Addr>=80h+01h=81h). És far en hexadecimal (0x)
  Serial1.write(0x4E); //S'envia un caràcter N en hexadecimal, tal i com el manual del SWIFT RAIL RS ens indica. (N caràcter = 78 decimal = 4E hexadecimal)
  Serial1.write(0x04);  //Envia la comanda que significa EOT (end of transmission).
  Serial1.flush();  //Aquesta comanda espera a que la comunicació sèrie finalitzi.
  digitalWrite(ReDe_Pin,LOW); //El pin de control de transmissió/recepció passa en estat baix, per tant, passarà a actuar com a "mestre" el SWIFT RAIL RS i com a "esclau" l'Arduino.
  delay(500);//S'espera 0,5 segons per assegurar que tot el procès ha estat realitzat al complet.

  if (!Serial1.available()) //En cas de que al port sèrie 1 no hi arribi res, entra en aquest condicional
  {
    Serial.println("no arriba res");  //Imprimeix per pantalla del pc
    delay(2000);  //S'espera 2 segons
  }
  else  //Si entra en aquest condicional significa que el serial té una informació preparada per a ser llegida.
  {
   Serial.println(""); //S'imprimeix per pantalla un espai en blanc, solament és per no ajutnar coses en l'impresió.
   Serial.println("Arriba informació...");  //Imprimeix per pantalla aquest text.

   while(Serial1.available())  // és mantindrà dins el bucle while mentres vagin arribant bytes d'informació resposta del SWIFT RAIL RS.
    {
      rebut[posicio]=Serial1.read();  //S'emmagatzemma el byte que arriba en cada volta del bucle al vector rebut, a la posició que li correspon.
      posicio=posicio+1; //A cada volta del bucle s'augmenta amb 1 unitat la posició del vector rebut.
    }
    delay (3000); //Espera 3 segons
    Serial.println("El valor del pes net és: ");  //Treu per pantalla el text

    for (posicio=3;posicio<12;posicio++)  //bucle que va de 3 a 11, serveix per definir les diferents posicions. Aquestes coincideixen amb les posicions del
    //vector rebut[] en les quals hi ha guardat els valors del Pes_Net, doncs són els que interessen.
   {
      pes[posicio-3]=rebut[posicio];  //A la posició que correspongui al vector pes[]de posició (0 a 8) se l'hi assigna el valor del vector rebut[] de posició (3-12).
      //pes[0]=direccio del port.
      //pes[1,2]=indica l'estat del pes: ("O"=pes major que la capacitat màxima, "S"=pes estable, "M"=pes no estable(en moviment), "E"=pes que no es pot detectar).
      //pes[3]=Dígit que dona les centenes de miler o signe negatiu.
      //pes[4]=Dígit decenes de miler.
      //pes[5]=Dígit miler.
      //pes[6]=Dígit centenes.
      //pes[7]=Dígit desenes.
      //pes[8]=Dígit unitats.
```

```
      Serial.print(pes[posicio-3]); //Imprimeix per pantalla del PC el valor que li correspon a cada posicio del vector pes[].
    }
    delay(3000);  //Espera 3 segons.
    Serial.println("");  //Treu per pantalla un espai per evitar ajuntar conceptes d'impressió.
    Serial.println("El valor que s'enviarà de pes és: "); //Treu per pantalla el text.

    for (posicio=2;posicio<9;posicio++) //El bucle va de la posició 2 a la 8, per tant agafa únicament les les posicions 2-8 del vector pes[].
    {
      pes_enviar[posicio-2]=pes[posicio]; //Guarda la variable de tipus char que hi havia en la posició corresponent d'acord amb l'índex, a un nou vector anomenat "pes_enviar".
     Serial.print(pes_enviar[posicio-2]);  //Imprimeix cada valor del nou vector de chars que enviarem. Aquest vector és el resultat ja filtrat de la sèrie de dades que donava la SWIFT RAIL RS.
    }
    if (pes_enviar[0]==0x2D)  //En cas que hi hagi un pes negatiu, el símbol - sempre està en la posicio 0 del vector "pes_enviar[]".
    //Com que és en char el negatiu(-) se compara amb la seva transcripció en hexadecimal, (char -> -= 0x 2D).
    {
      Serial.println(""); //Imprimeix per pantalla un espai.
      Serial.println("Pes negatiu...no tenir en compte la dada, hi ha algun error!"); //Imprimeix per pantalla el text.
    }
    delay(3000);  //Espera 3 segons.
  }
  return pes_enviar;
}

void Lectura_dades_Radar()
{
 #define PI 3.14159

  int temps=0;  //variable on es guarda el valor de temps que la senyal de so torna a ser rebuda.
  float distancia;  //defineix la variable distància, és on es guardarà el valor de la distància entre sensor i objecte.
  float c=343.2;  //defineix la variable c, que fa referència a la velocitat de les ones en el medi de transmissió.[m/s]
  float h1=635;  //mm
  float h2=2424; //mm
  float L=4500; //mm
  float D=3400; //mm
  float d=600;  //mm
  float densitat=600; //Kg/m^3
  float Volum_restant=0;
  float pes_enviar2=0;
  int pes[7]; //vector en enters utlitzat per la transformació del valor float de distància a un vector de char.
  int i;  //variable utilitzada per recorrer el vector pes[].
```

```
  h1=h1/1000;
  h2=h2/1000;
  L=L/1000;
  D=D/1000;
  d=d/1000;

  digitalWrite(Pin_triger,LOW);  //serveix per forçar el pin de sortida triger estar en estat baix o 0 lògic en el moment d'inicialització.
  digitalWrite(Pin_triger,HIGH); //Posa el pin digital numero 3 una senyal "high" o 1 lògic.
  delayMicroseconds(10);  //Envia una senyal "high" en el pin 3 durant 10 microsegons.
  digitalWrite(Pin_triger,LOW); //Posa el pin digital número 3 com a senyal "low" o 0 lògic.
  temps=pulseIn(Pin_echo,HIGH); //Aquesta comanda de pulseIn(pin,value) serveix per contar el temps en microsegons entre quan està en estat "high" o "low".
  distancia=c*(temps*0.000001)/2; //calcul de la distància d'acord amb el procediment ja explicat en la teoria.
  if (distancia<0)  //Algunes que altres vegades hi ha algun error en la mesura del temps hi surt negativa, en aquest cas entra aquí i no fa el càlcul de distància ja.
  {
    Serial.println("ERROR!");
  }
  else  //En cas que la distància sigui un valor lògic, és a dir positiu, entra en aquest bucle.
  {
   if (distancia >= h1+L)
    {
     Volum_restant=((PI*(L+h1+h2-distancia))/12)*(D*D+D*d+d*d);
    }
    else if(distancia >= h1 && distancia < h1+L)
    {
     Volum_restant=((PI*h2)/12)*(D*D+D*d+d*d)+((PI*D*D)/4)*(L+h1-distancia);
    }
    else
    {
    Volum_restant=((PI*h2)/12)*(D*D+D*d+d*d)+((PI*D*D)/4)*L+((PI*(h1-distancia))/12)*(D*D+D*d+d*d);
    }

    pes_enviar2=Volum_restant*densitat;
    Serial.print("El temps és: " ); //Imprimeix per pantalla aquest text.
    Serial.print(temps);  //Imprimeix per pantalla tot seguit a la comanda anterior el temps entre pulsos
    Serial.print(" microsegons.  ");  //Imprimeix per pantalla tot seguit a la comanda anterior i en acabar salta de línia.
    Serial.print ("I la distància és de: ");
    Serial.print(distancia);
    Serial.print(" m.  ");
    Serial.print(" I la massa restant és: ");
    Serial.println(pes_enviar2);
  }
  delay(5000);  //Espera 5 segons entre volta i volta
```

```
  //Es tracta el valor float del pes per passar-lo amb vector de char, que és el format que necesita el mòdul per enviar dades.
  //El procediment de transformar els valor enter en vector d'enters és el següent:
  pes[0]=0;
  pes[1]=pes_enviar2/100000;
  pes[2]=(pes_enviar2-pes[1]*100000)/10000;
  pes[3]=(pes_enviar2-pes[1]*100000-pes[2]*10000)/1000;
  pes[4]=(pes_enviar2-pes[1]*100000-pes[2]*10000-pes[3]*1000)/100;
  pes[5]=(pes_enviar2-pes[1]*100000-pes[2]*10000-pes[3]*1000-pes[4]*100)/10;
  pes[6]=(pes_enviar2-pes[1]*100000-pes[2]*10000-pes[3]*1000-pes[4]*100-pes[5]*10);

 for (i=0;i<7;i++)  //A cada valor del vector pes, se li assigna un caràcter en hexadecimal, per tal de fer la transformació de enter a char.
  {
    if (pes[i]==0)
    {
      pes_enviar[i]=0x30; //equivalent al caràcter 0.
    }
    else if(pes[i]==1)
    {
      pes_enviar[i]=0x31; //equivalent al caràcter 1.
    }
     else if(pes[i]==2)
    {
     pes_enviar[i]=0x32; //equivalent al caràcter 2.
    }
     else if(pes[i]==3)
    {
      pes_enviar[i]=0x33; //equivalent al caràcter 3.
    }
     else if(pes[i]==4)
    {
      pes_enviar[i]=0x34; //equivalent al caràcter 4.
    }
     else if(pes[i]==5)
    {
      pes_enviar[i]=0x35; //equivalent al caràcter 5.
    }
     else if(pes[i]==6)
    {
      pes_enviar[i]=0x36; //equivalent al caràcter 6.
    }
     else if(pes[i]==7)
    {
      pes_enviar[i]=0x37; //equivalent al caràcter 7.
    }
     else if(pes[i]==8)
    {
```

```
      pes_enviar[i]=0x38; //equivalent al caràcter 8.
    }
     else if(pes[i]==9)
    {
      pes_enviar[i]=0x39; //equivalent al caràcter 9.
    }
  }
  return pes_enviar;
}

void Enviar_dades()
{
  Serial.println("");
  Serial.println("Provant la comunicació entre el SIM900 i l'Arduino");
  SIM_card.println("AT");      //la resposta esperada és "OK"
  delay(1000);
  printSerialData();     //executa la funció que treu per pantalla pc la resposta del mòdul a les
comandes AT
  delay(1000);
  Serial.println("------------------------------");

  Serial.println("Establint connexió amb l'APN...");
  SIM_card.println("AT+CSTT=\"movistar.es\",\"\",\"\""); //Retornarà
'AT+CSTT='movistar.es','','','OK'
  delay(5000);
  printSerialData();     //executa la funció que treu per pantalla pc la resposta del mòdul a les
comandes AT
  delay(1000);
  Serial.println("------------------------------");

  Serial.println("Iniciar la connexió sense cable...");
  SIM_card.println("AT+CIICR");     //la resposta esperada és "OK"
  delay(5000);
  printSerialData();     //executa la funció que treu per pantalla pc la resposta del mòdul a les
comandes AT
  delay(2000);
  Serial.println("");
  Serial.println("------------------------------");

  Serial.println("Retorna la IP local");
  SIM_card.println("AT+CIFSR");     //la resposta esperada és la ip del mòdul
  delay(5000);
  printSerialData();     //executa la funció que treu per pantalla pc la resposta del mòdul a les
comandes AT
  delay(1000);
  Serial.println("");
  Serial.println("------------------------------");

  Serial.println("Connectant amb el servidor i el port determinat...");
```

```
  SIM_card.println("AT+CIPSTART=\"TCP\",\"test.mosquitto.org\",\"1883\"");
//Ha de retornar "CONNECT OK"
  delay(5000);
  printSerialData();      //executa la funció que treu per pantalla pc la resposta del mòdul a les
comandes AT
  delay(3000);
  Serial.println("");
  Serial.println("------------------------------");

  Serial.println("Enviant CONNECT i PUBLISH al servidor...");
  SIM_card.println("AT+CIPSEND");          //la resposta esperada és >..dades i finalment
"SEND OK"
  delay(5000);
  printSerialData();      //executa la funció que treu per pantalla pc la resposta del mòdul a les
comandes AT
  delay(1000);
  enviar_connect();      //executa la funció enviar_connect()
  delay(1000);
  enviar_publish();      //executa la funció enviar_publish()
  delay(5000);
  printSerialData();      //executa la funció que treu per pantalla pc la resposta del mòdul a les
comandes AT
  Serial.println("");
  Serial.println("");
  Serial.println("------------------------------");

  Serial.println("Tancant la connexió del port...");
  SIM_card.println("AT+CIPSHUT");          //La resposta esperada es "SHUT OK"
  delay(5000);
  printSerialData();
  delay(2000);
  Serial.println("");
  Serial.println("------------------------------");
}

void printSerialData() //Aquesta funció serveix per imprimir per pantalla la resposta de la comanda
AT introduida, i guarda aquesta en la variable "resposta"
{
  while (SIM_card.available()!=0) //Mentres la SIM card estigui funcionant
    {
     Serial.write(SIM_card.read()); //Escriure per pantalla en format binari la informació que arriba
de la SIM card, que serà la resposta a les comandes AT.
    }
}

void enviar_connect() //Funció que serveix per enviar el vector "connect".
{
  int i;  //Variable utilitzada per recòrrer els components del vector con[50].
```

```
  char con[50]={0x10,0x12,0x00,0x04,0x4D,0x51,0x54,0x54,0x04,0x02,0x00,0x3C,0x00,0x06,0x41,0x42,0x43,0x44,0x45,0x46}; //Vector connect que conté els components en hexadecimal que són necessàris, ja explicat en la part teòrica.

  for (i=0;i<20;i++)  //Bucle que recorre tots els components del vector con[50] que estàn escrits, i els imprimeix per pantalla.
  {
    SIM_card.print(con[i]);
  }
}

void enviar_publish() //Funció que serveix per enviar el vector pub_enviar[50]
{
  char stat[10]={0x2C,0x22,0x73,0x74,0x61,0x74,0x75,0x73,0x22,0x3A};  //vector de tipus char que hi ha : ",”status”:"
  char pes_igual[6]={0x22,0x70,0x65,0x73,0x22,0x3A};  //vector de tipus char que hi ha: ""pes":"
  char pub_enviar[50];  //Vector de 50 posicions perquè el número de posicions necesàries a enviar dades no es constant, pot variar en un futur.
  int i;  // Variable per recorrer les posicions dels vectors.

  longitud_pub(); //Aplica la funció longitud_pub();
  long_granja();  //Aplica la funció long_granja();
  for (i=0;i<longitud;i++)     //Recorrer el vector pub_enviar[50] des de la posició 0 fins a la longitud del vector pub -1.
  {
    if (i!=1)   // Si la posició és diferent de la 1, iguala el vector "pub_enviar[]" amb el "pub[]" perquè els valors aquests seràn sempre constants.
    {
      pub_enviar[i]=pub[i];
    }
    else  //Si la posició és 1, pertany a la longitud total del missatge, que serà els 26 components constants del vector pub[], juntament amb les 6 posicions que ocupa
          // el vector de retorn de pes del SWIFT RAIL RS.
    {
      pub_enviar[i]=27+7+longitud_granja+1+1+6+10; //els 27 és la longitdu constant del vector inicial pub[], els 7 són de la mesura, la longitud_granja és la llargada del nom de la granja, i el els dos 1 són de la "," i "}",
 //el 6 és del text pes, i els 10 del text status.
    }
  }
  for (i=0;i<longitud;i++)    //Envia al mòdul de la SIM card, els valors del vector pub_enviar[] creat anteriorment, fins a la posició que pertany a la longitud ja mesurada anteriorment també.
  {
    SIM_card.print(pub_enviar[i]);
  }
  for (i=0;i<longitud_granja;i++)   //Envia al mòdul de la SIM card,els valors del vector granja[].
  {
    SIM_card.print(granja[i]);
```

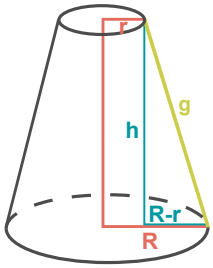
```
  }
  SIM_card.print(","); //Envia al mòdul de la SIM card, la ",".
  for (i=0;i<6;i++) //Envia a la SIM card el nom de pes.
  {
    SIM_card.print(pes_igual[i]);
  }
  for (i=1;i<7;i++) //Envia al mòdul de la SIM card, els valors que llegeix el SWIFT RAIL RS.
  {
   SIM_card.print(pes_enviar[i]);
  }
  for (i=0;i<10;i++)  //Envia a la SIM card el text d'status.
  {
    SIM_card.print(stat[i]);
  }
  SIM_card.print(pes_enviar[0]);  //Envia a SIM card l'estat.
  SIM_card.print("}");  //Envia una corxeta per finalitzar l'estil Json.
  SIM_card.write(0x1A); //Envia un cntrl+z...que es igual a enviar en format hexadecimal.
  delay(2000);  //Espera uns 2 segons.
}

void longitud_pub() //Funció que llegeix la longitud de vector ocupada per caràcters hexadecimals del vector pub[50].
{
  int cont_zero=0;  //variable que contarà el 0 que hi hagi en el vector pub[50], és a dir els llocs buits.
  int i;  //variable utilitzada per avançar a l'interior del vector pub[50].
  for (i=0;i<50;i++) //Recorre tot el vector des de la posició 0 a la 49.
  {
    if (pub[i]==0)  //Si troba un 0, entra al condicional.
    {
      cont_zero++;  //Augmenta amb 1 unitat el valor de la variable "cont_zero".
    }
  }
  longitud=50-cont_zero+1;  //Emmagatzema la diferència entre les 50 posicions inicials, el valor de 0 trobats...i se l'hi suma un que va per defecte en el vector pub[50] ja.
}

void long_granja()  //Funció que calcula la longitud del vector que conté el nom de la granja.
{
  int i;  //Variable que serveix per recórrer tot el vector granja[].
  int cont_nul=0;
  for (i=0;i<20;i++)
  {
   if (granja[i]==0) //Si el component i del vector granja es nul, es ha dir espai en blanc, s'augmenta el contador.
    {
      cont_nul++;
    }
  }
```

```
  longitud_granja = 25 - cont_nul; //La longitud del vector serà el numero màx de posicions, 25 menos el nombre d'espais en blanc.
}
```

14.2. Càlcul de volum del silo:



Il·lustració 55: Tronc de con.

El volum del tronc de con segueix la següent equació:

$$V=\frac{\pi\cdot h}{3}\cdot[R^2+R\cdot r+r^2]$$

D'acord amb aquesta es calcularan tots els volums de la part 1 i 3 del silo.

$$V_{1\,Total}=\frac{\pi\cdot h_1}{3}\cdot[(\frac{D}{2})^2+\frac{D}{2}\cdot\frac{d}{2}+(\frac{d}{2})^2]=\frac{\pi\cdot h_1}{3}\cdot[\frac{D^2+D\cdot d+d^2}{4}]=\frac{\pi\cdot h_1}{12}\cdot[D^2+D\cdot d+d^2]$$

$$V_{3\,Total}=\frac{\pi\cdot h_2}{12}\cdot[D^2+D\cdot d+d^2]$$

I pel que fa al cilindre de la part 2:

$$V_{2\,Total}=\pi\cdot(\frac{D}{2})^2\cdot L=\frac{\pi\cdot D^2}{4}\cdot L$$

Doncs bé, si es vol tenir el volum en funció de la distància entre el sensor i el pinso cal contemplar les 3 zones i simplement substituir la altura del recipient per la seva expressió concreta.

- Si es troba en el tram 1:

$$h=h_1-y$$

El volum total serà

$$V_T=V_{3\,Total}+V_{2\,Total}+\frac{\pi\cdot(h_1-y)}{12}\cdot[D^2+D\cdot d+d^2]$$

- Durant el tram 2:

$$h=L+h_1-m$$

El volum total és:

$$V_T=V_{3\,Total}+\frac{\pi\cdot D^2}{4}\cdot(L+h_1-m)$$

- Si és en el 3er tram:

$$h=L+h_1+h_2-m$$

$$V_T=\frac{\pi\cdot(L+h_1+h_2-m)}{12}\cdot[D^2+D\cdot d+d^2]$$